Computer Vision
- Feature Descriptors & Matching

Junjie Cao @ DLUT

Spring 2019

What we will learn today?

- SIFT: an image region descriptor
- HOG: another image descriptor
- Other image descriptors

- Matching
- Application: Panorama

1999年British Columbia大学大卫.劳伊（David G.Lowe）教授总结了现有的基于不变量技术的特征检测方法，并正式提出了一种基于尺度空间的、对图像缩放、旋转甚至仿射变换保持不变性的图像局部特征描述算子－SIFT（尺度不变特征变换，Scale Invariant Feature Transform），这种算法在2004年被加以完善。

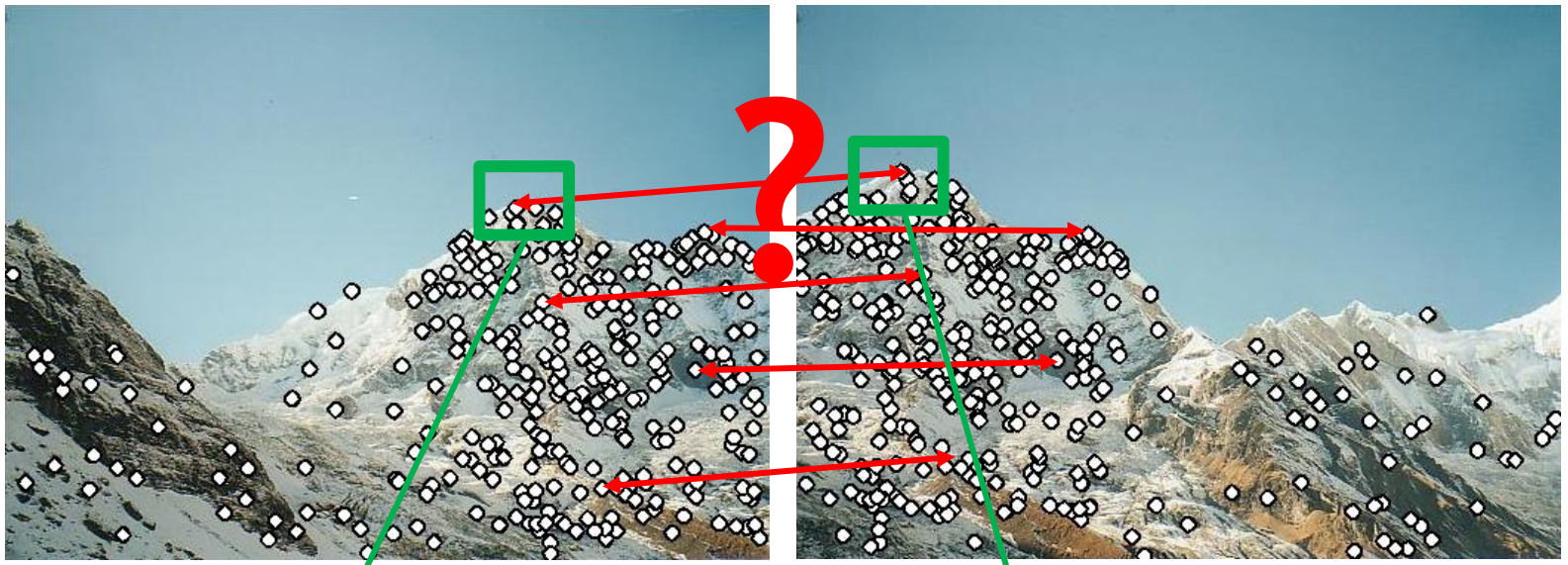
Some background reading: David Lowe, IJCV 2004

Local Descriptors

- We know how to detect points
- Next question:

How to *describe* them for matching?

Slide credit: Kristen Grauman



$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$ x1

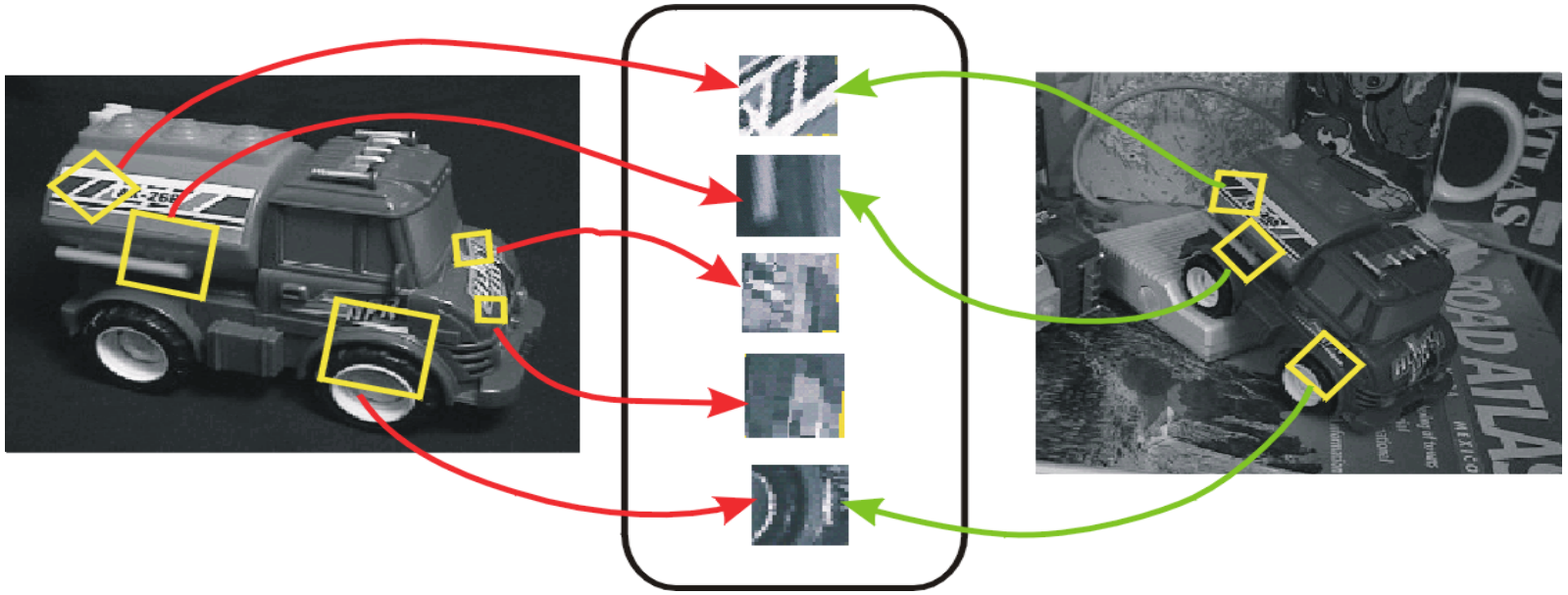
$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$ x2

Point descriptor should be:

1. **Invariant**
2. **Distinctive**

Invariant Local Features

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



Following slides credit: CVPR 2003 Tutorial on **Recognition and Matching Based on Local Invariant Features** David Lowe

Advantages of invariant local features

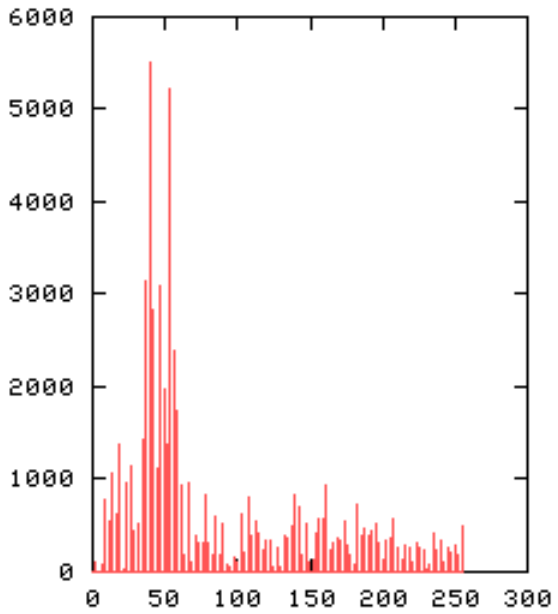
- Locality: features are local, so robust to occlusion and clutter (no prior segmentation)
- Distinctiveness: individual features can be matched to a large database of objects
- Quantity: many features can be generated for even small objects
- Efficiency: close to real-time performance
- Extensibility: can easily be extended to wide range of differing feature types, with each adding robustness

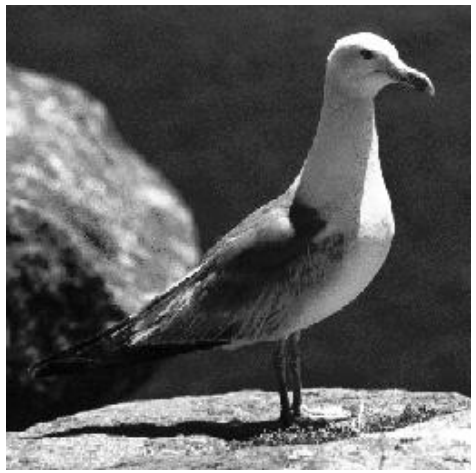
Image representations

- Templates
  - Intensity, gradients, etc.

- Histograms
  - Color, texture, SIFT descriptors, etc.

Image Representations: Histograms





Global histogram to represent distribution of features

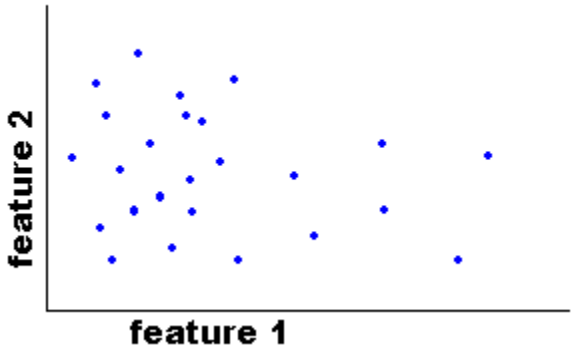
- Color, texture, depth, …

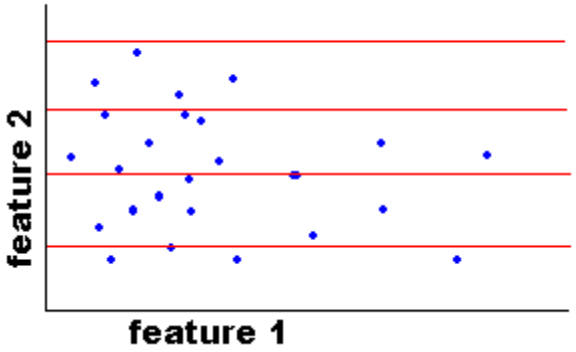
Local histogram per detected point

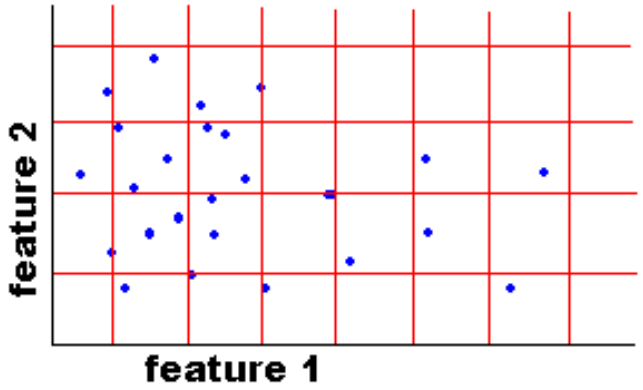
Images from Dave Kauchak

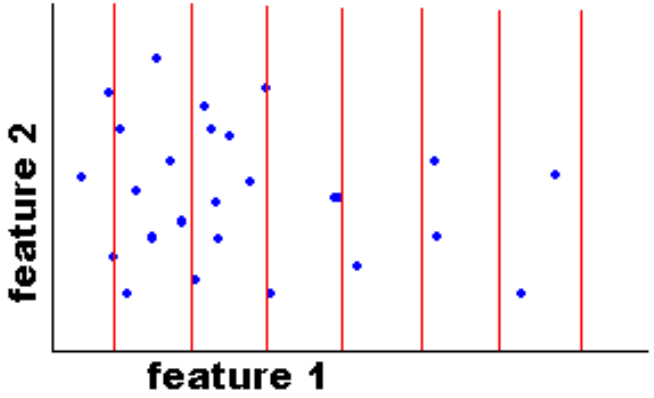
Image Representations: Histograms

Histogram: Probability or count of data in each bin









- Joint histogram
  - Requires lots of data
  - Loss of resolution to avoid empty bins

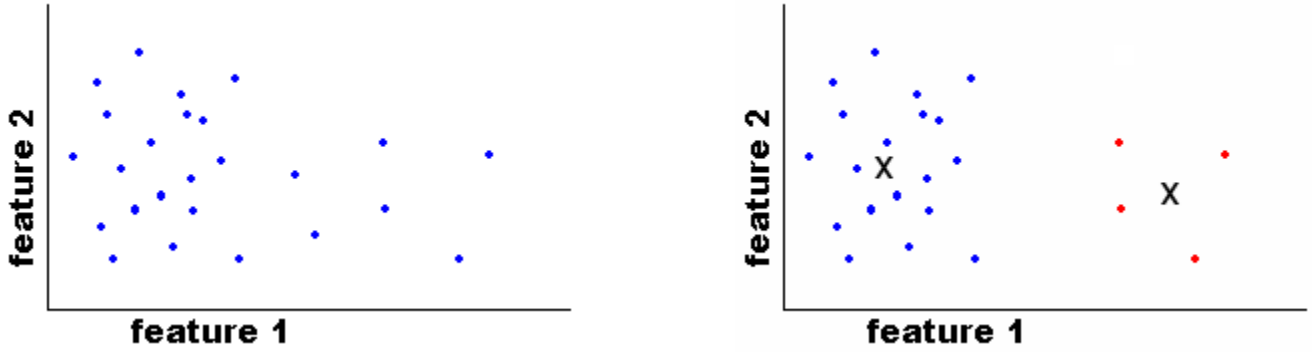
Marginal histogram

- Requires independent features
- More data/bin than joint histogram

Images from Dave Kauchak

Image Representations: Histograms

Clustering



Use the same cluster centers for all images

Images from Dave Kauchak

Computing histogram distance

$$\text{histint}(h_i, h_j) = 1 - \sum_{m=1}^{K} \min\left(h_i(m), h_j(m)\right)$$

Histogram intersection (assuming normalized histograms)

$$\chi^2(h_i, h_j) = \frac{1}{2} \sum_{m=1}^{K} \frac{[h_i(m) - h_j(m)]^2}{h_i(m) + h_j(m)}$$

Chi-squared Histogram matching distance

Cars found by color histogram matching using chi-squared

Histograms: Implementation issues

- Quantization
  - Grids: fast but applicable only with few dimensions
  - Clustering: slower but can quantize data in higher dimensions

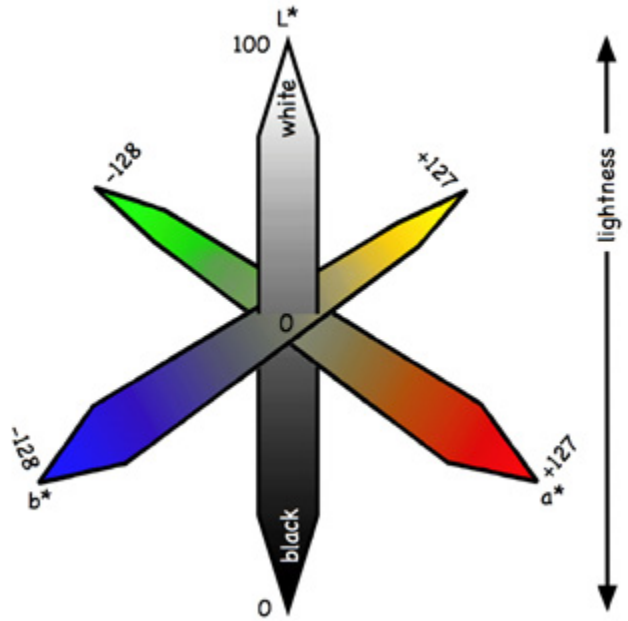
Few Bins
Need less data
Coarser representation

Many Bins
Need more data
Finer representation

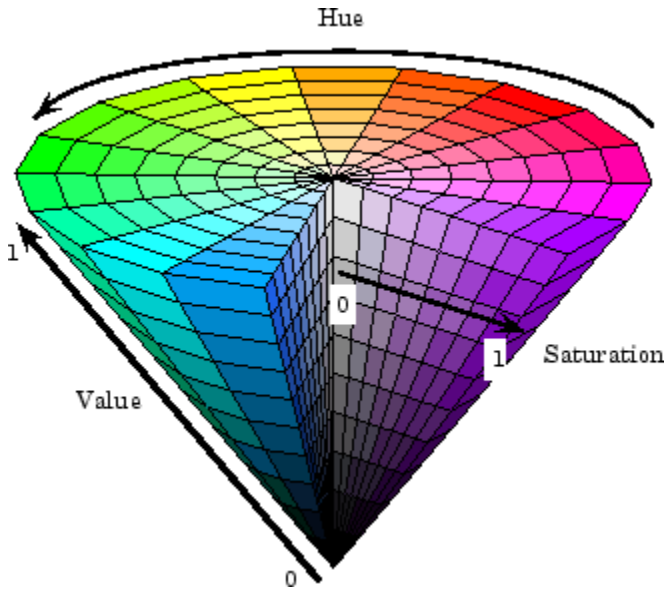
- Matching
  - Histogram intersection or Euclidean may be faster
  - Chi-squared often works better
  - Earth mover's distance is good for when nearby bins represent similar values

For what things might we compute histograms?

- Color



L*a*b* color space

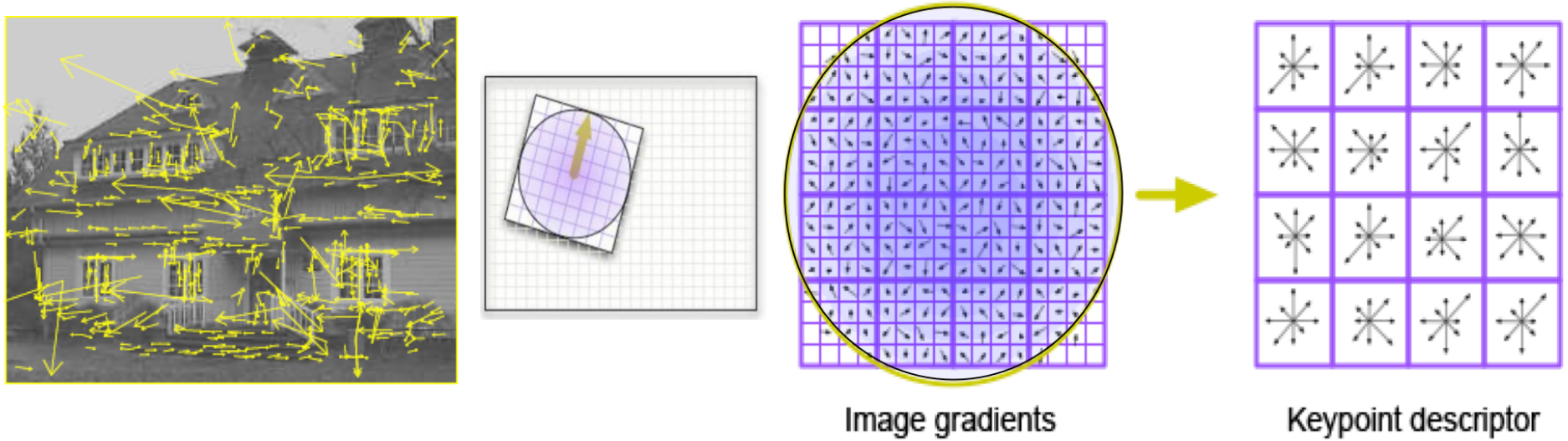


HSV color space

- Model local appearance

For what things might we compute histograms?

- Texture
- Local histograms of oriented gradients
- SIFT: Scale Invariant Feature Transform
  - Extremely popular (40k citations)

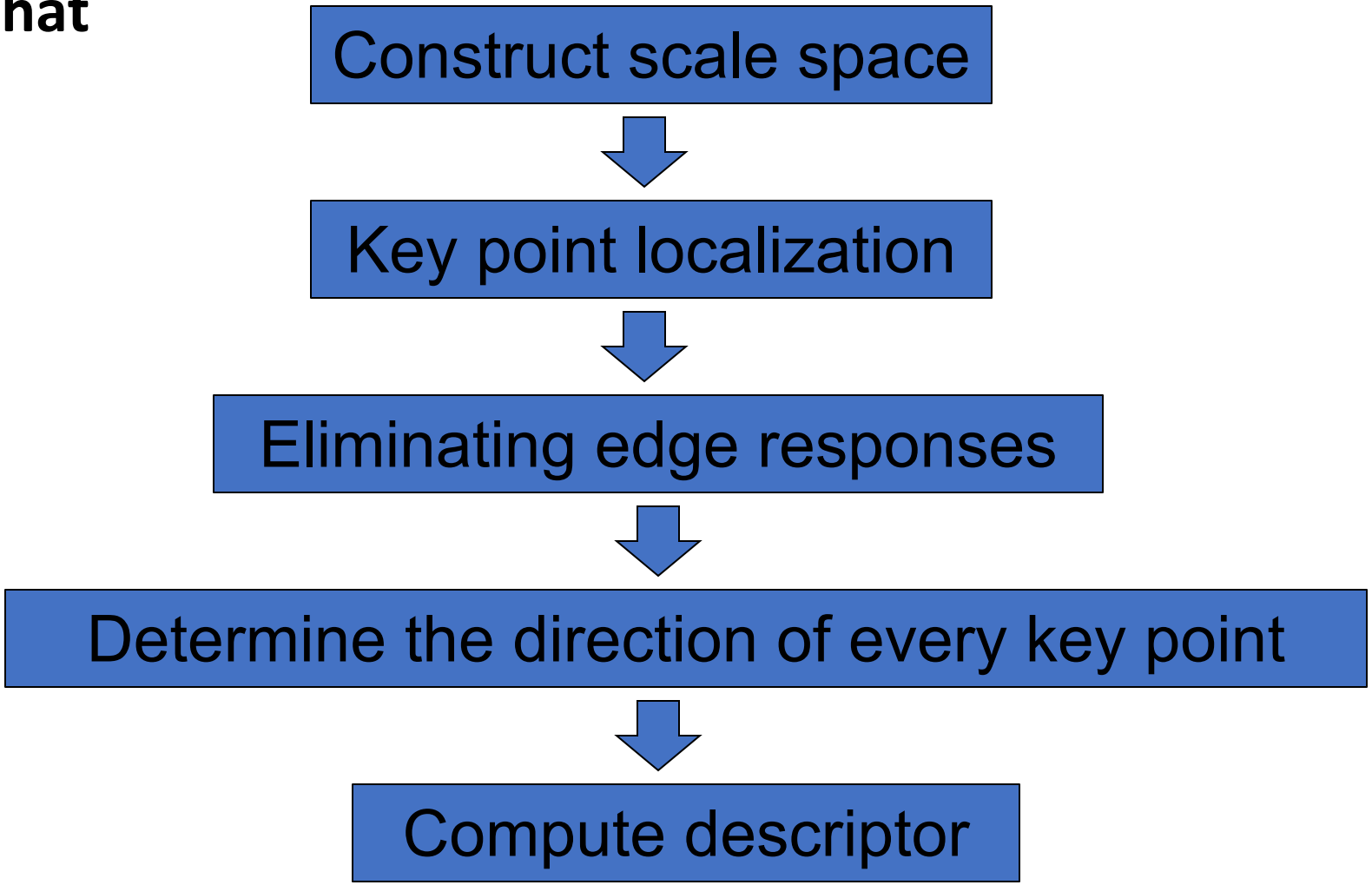


SIFT – Lowe IJCV 2004

SIFT processing

- Find Difference of Gaussian scale-space extrema as feature point locations
- Post-processing
  - Subpixel position interpolation
  - Discard low-contrast points
  - Eliminate points along edges
- Orientation estimation per feature point
- Descriptor extraction
  - Motivation: We want some sensitivity to spatial layout, but not too much, so blocks of histograms give us that.

SIFT flowchat

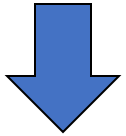


Key point localization

在计算过程中，分别对图像的行、列及尺度三个量进行了修正， 其修正结果如下：

$$D(X+\Delta X)=D(X)+(\frac{\partial D}{\partial X})^T(\Delta X)+\frac{1}{2}X^T\frac{\partial^2 D}{\partial X^2}\Delta X,\quad X=(x,y,\sigma)$$

$$\min\{D(X+\Delta X)-D(X)\}=\min\{(\frac{\partial D}{\partial X})^T\Delta X+\frac{1}{2}(\Delta X)^T\frac{\partial^2 D}{\partial X^2}(\Delta X)\}$$



$$\Delta X=-(\frac{\partial D}{\partial X})^T(\frac{\partial^2 D}{\partial X^2})^{-1}$$

Compute: $D(X+\Delta X)=D(X)+\frac{1}{2}(\frac{\partial D}{\partial X})^T\Delta X$

$D(X+\Delta X)>t$  extreme point

Eliminating edge responses

The eigenvalues of H are proportional to the principal curvatures of D, we can avoid explicitly computing the eigenvalues, as we are only concerned with their ratio. Let α be the eigenvalue with the largest magnitude and β be the smaller one

$$Tr(H) = D_{xx} + D_{yy} \qquad Det(H) = D_{xx} \times D_{yy} - D_{xy} \times D_{xy}$$

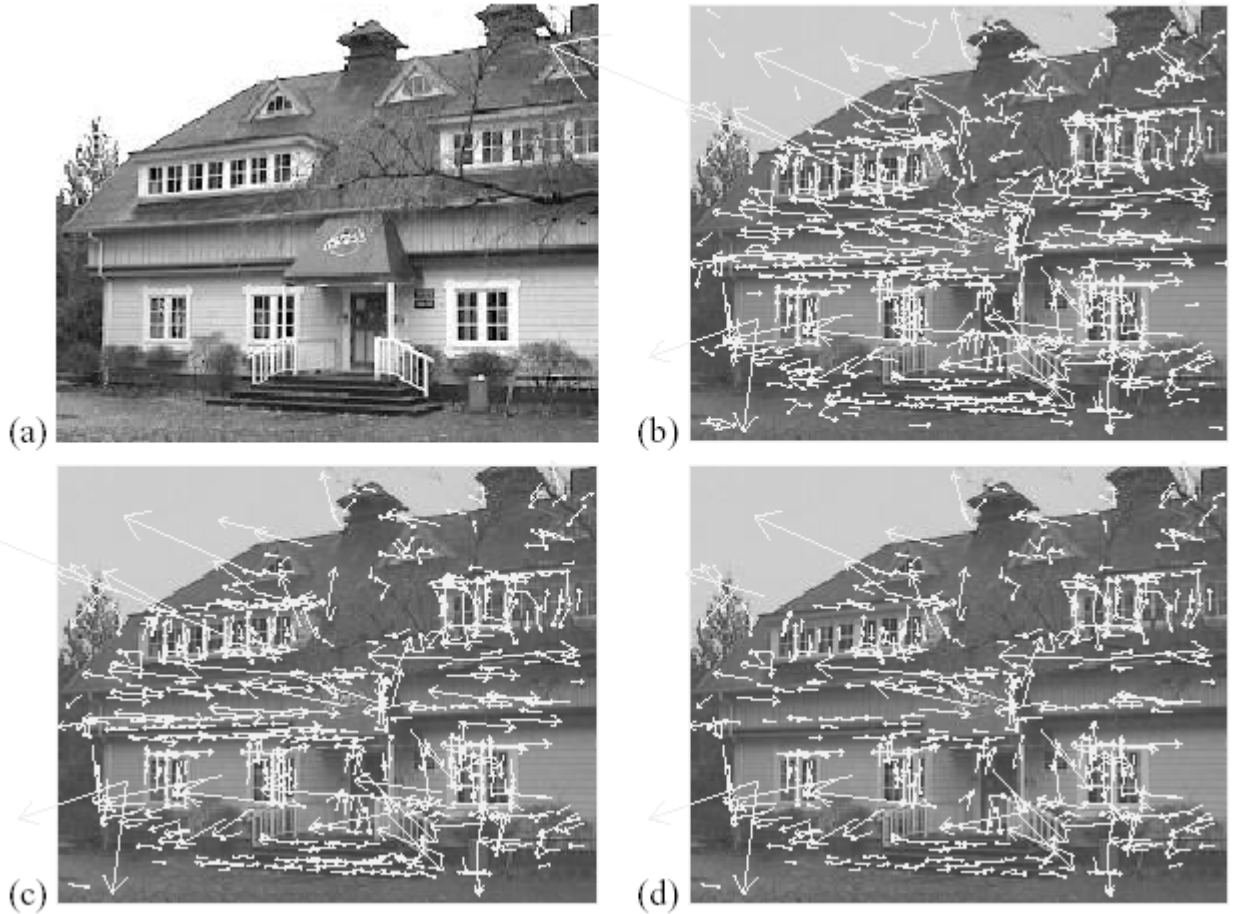
$$\alpha = r\beta \qquad \frac{Tr(H)^2}{Det(H)} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r+1)^2}{r}$$

rejecting keypoints for which $\frac{Tr(H)^2}{Det(H)} > \frac{(r+1)^2}{r}$

(r=10 in Lowe)

Example of keypoint detection

Threshold on value at DOG peak and on ratio of principle curvatures (Harris approach)



(a) 233x189 image
(b) 832 DOG extrema
(c) 729 left after peak value threshold
(d) 536 left after testing ratio of principle curvatures

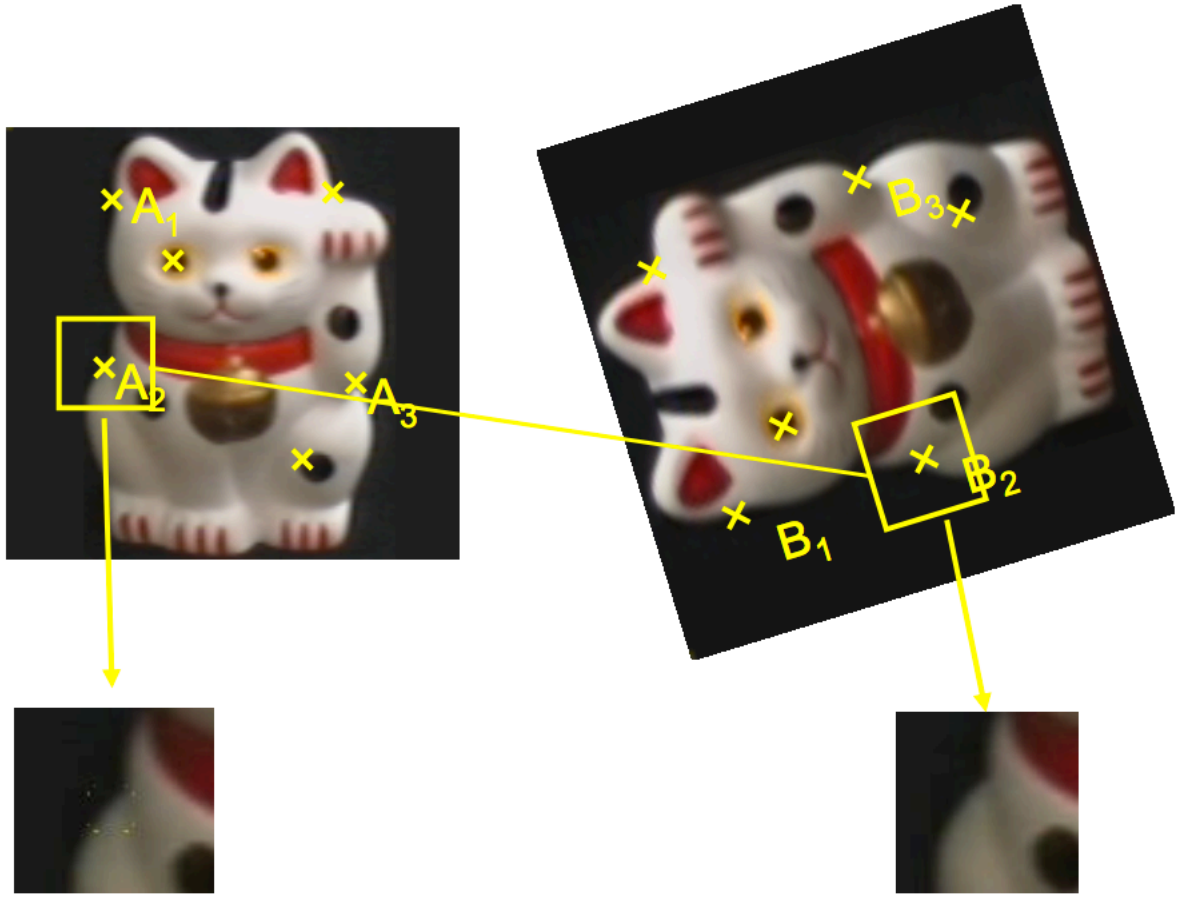
Vectors indicate scale, orientation and location.

Scale invariance

Requires a method to repeatably select points in location and scale:

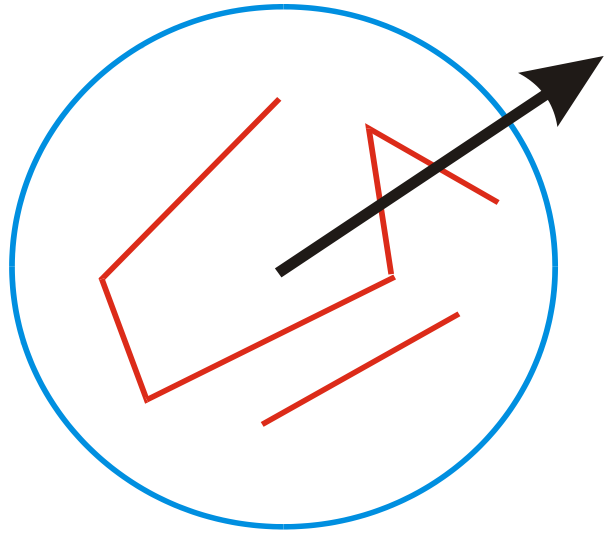
- The only reasonable scale-space kernel is a Gaussian (Koenderink, 1984; Lindeberg, 1994)
- An efficient choice is to detect peaks in the difference of Gaussian pyramid (Burt & Adelson, 1983; Crowley & Parker, 1984 – but examining more scales)
- Difference-of-Gaussian with constant ratio of scales is a close approximation to Lindeberg's scale-normalized Laplacian (can be shown from the heat diffusion equation)

Becoming rotation invariant



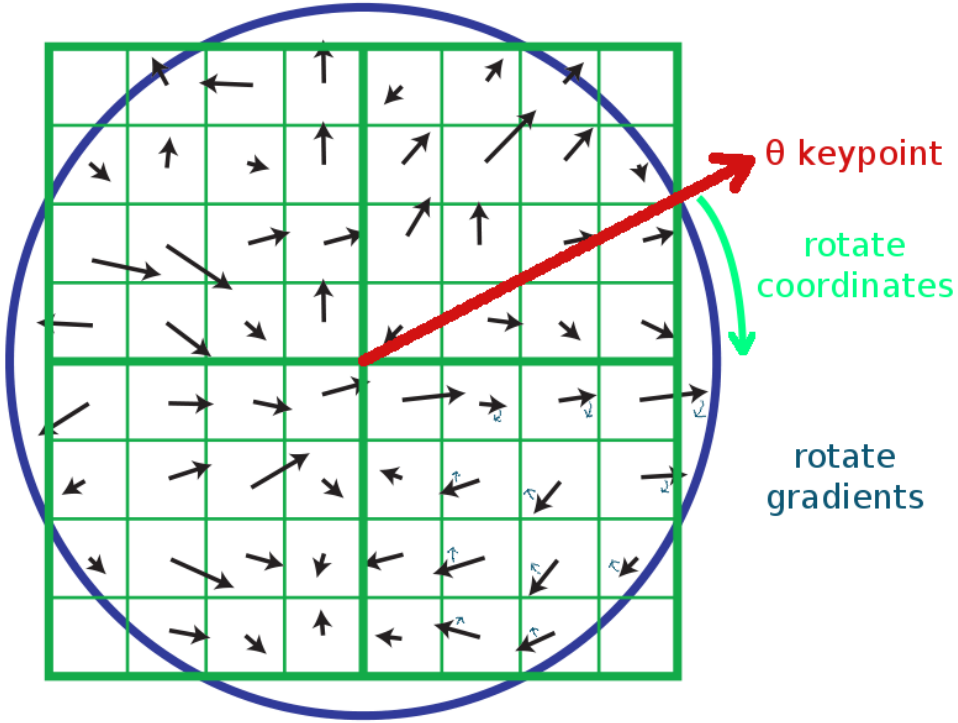
Becoming rotation invariant

- We are given a keypoint and its scale from DoG

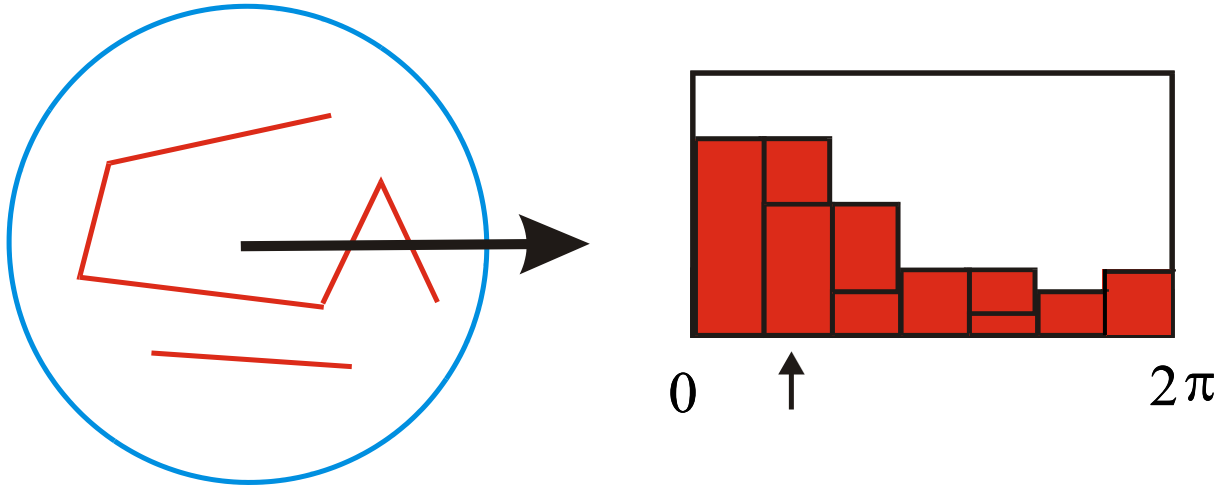


- We will describe all features **relative** to this orientation
- Causes features to be rotation invariant!
  - If the keypoint appears rotated in another image, the features will be the same, because they're **relative** to the characteristic orientation

SIFT descriptor formation



- Use the blurred image associated with the keypoint's scale
- Compute gradient orientation histogram over the keypoint neighborhood.
- Select dominant orientation Θ



- Rotate the gradient directions AND locations by -Θ
  - Now we've cancelled out rotation and have gradients expressed **relative** to keypoint orientation θ
  - We could also have just rotated the whole image by -θ, but that would be slower.

Determine the direction of every key point

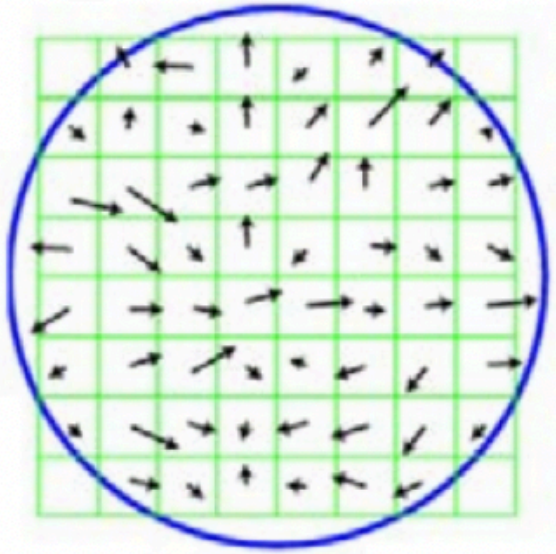


Image gradients

$$radius = \frac{3\sigma_{oct} \times \sqrt{2} \times (d+1) + 1}{2} \qquad d = 4$$

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

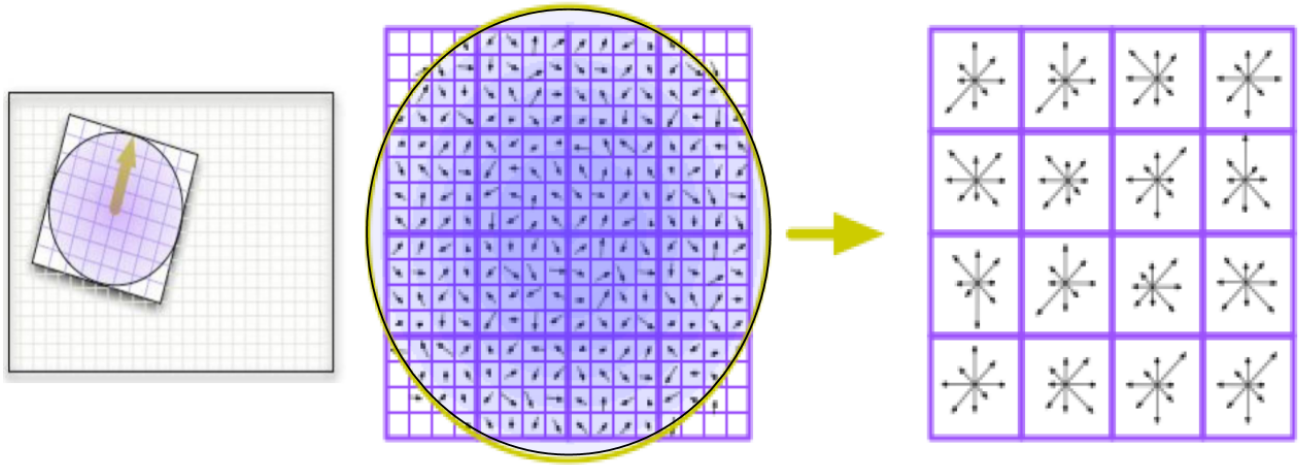
$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

SIFT processing

- Find Difference of Gaussian scale-space extrema as feature point locations
- Post-processing
  - Subpixel position interpolation
  - Discard low-contrast points
  - Eliminate points along edges
- Orientation estimation per feature point
- Descriptor extraction
  - Motivation: We want some sensitivity to spatial layout, but not too much, so blocks of histograms give us that.

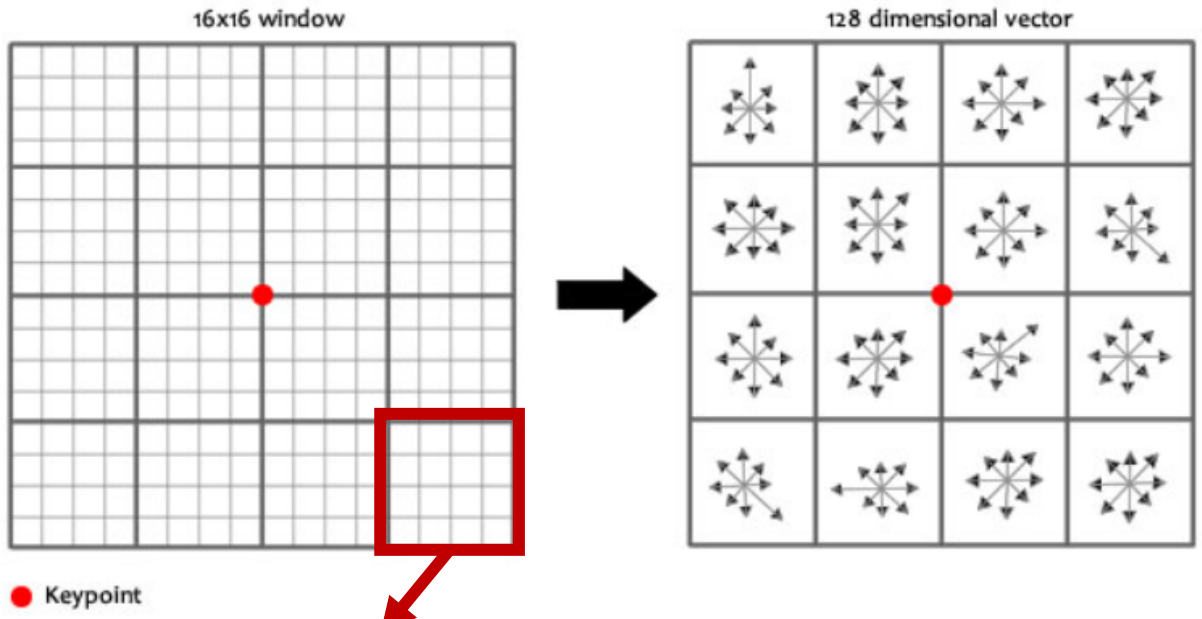
SIFT Descriptor Extraction

- Given a keypoint with scale and orientation:
  - Pick scale-space image which most closely matches estimated scale
  - Resample image to match orientation OR
  - Subtract detector orientation from vector to give invariance to general image rotation.

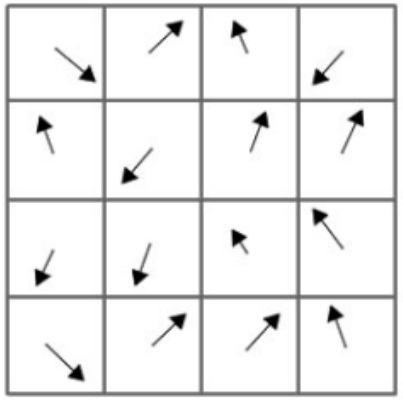


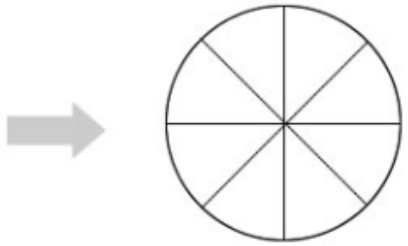
SIFT Descriptor Extraction

- Given a keypoint with scale and orientation



Gradient magnitude and orientation





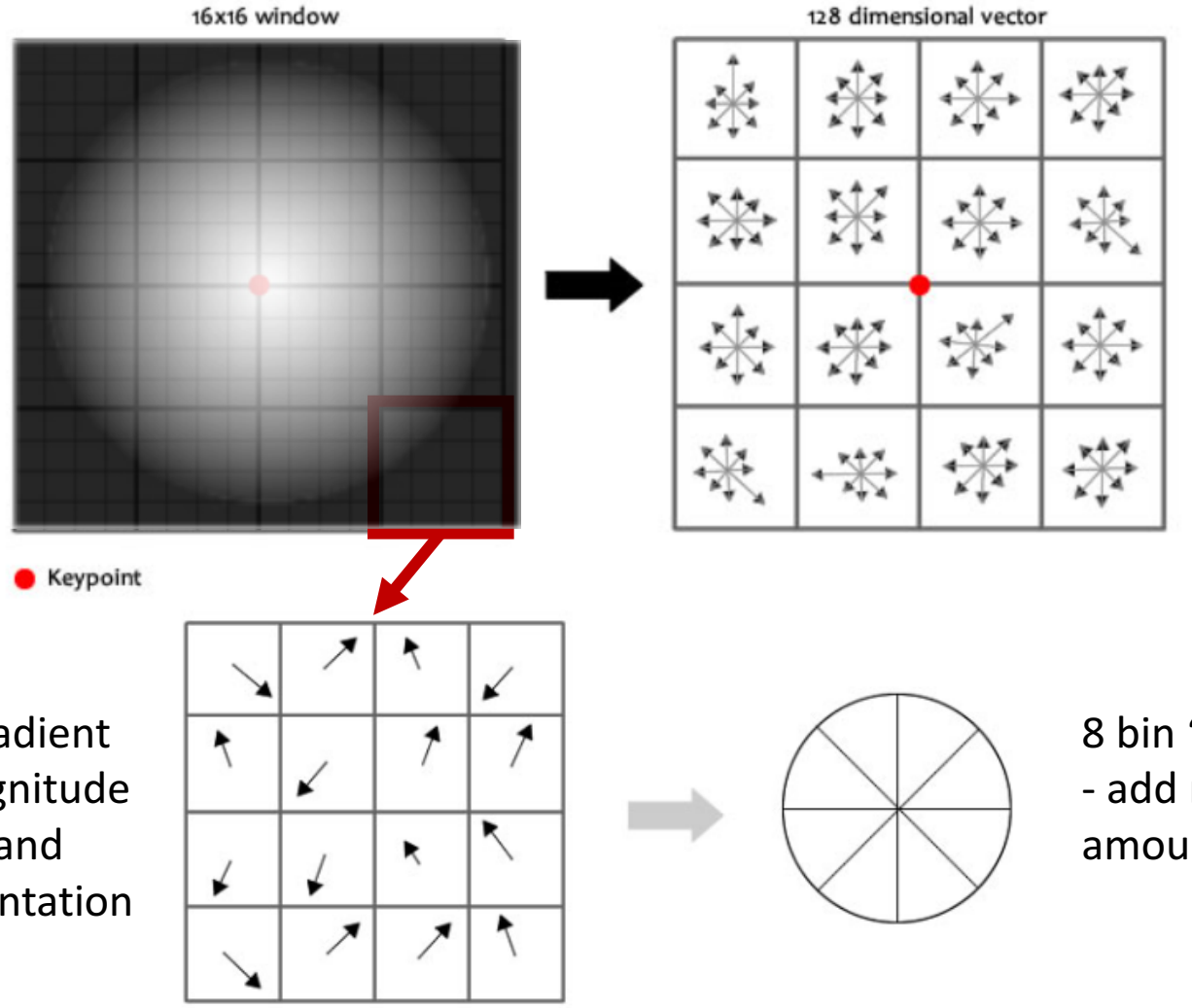
8 bin ‘histogram’
- add magnitude amounts!

Utkarsh Sinha

SIFT Descriptor Extraction

Weight 16x16 grid by Gaussian to add location robustness and reduce effect of outer regions

σ = half window width



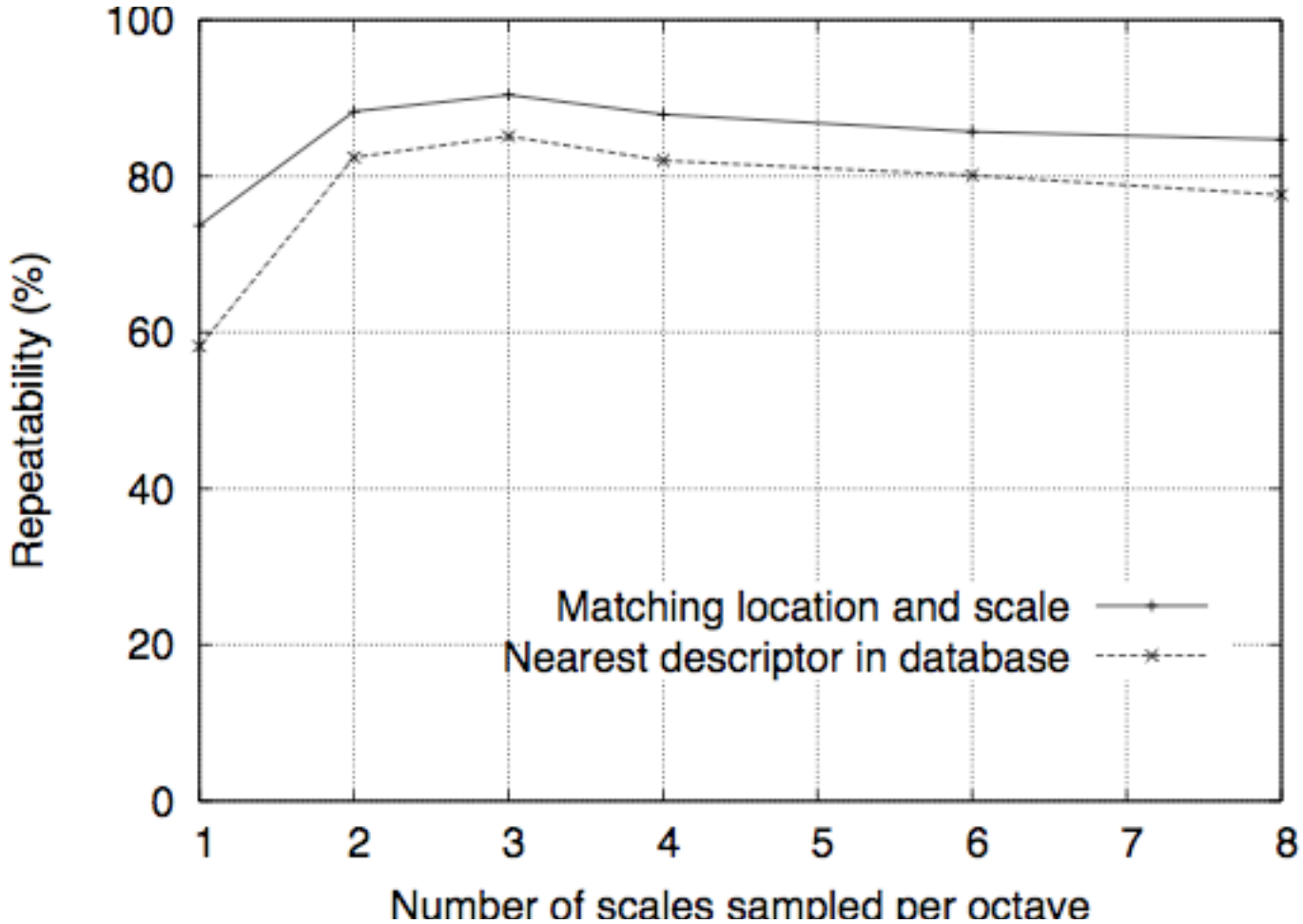
Utkarsh Sinha

SIFT Descriptor Extraction

- Extract 8 x 16 values into 128-dim vector
- Illumination invariance:
  - Working in gradient space, so robust to $I = I + b$
  - Normalize vector to [0…1]
    - Robust to $I = \alpha I$ brightness changes
  - Clamp all vector values > 0.2 to 0.2.
    - Robust to “non-linear illumination effects”
    - Image value saturation / specular highlights
  - Renormalize
  - Result is a vector which is fairly invariant to illumination changes.

Specular highlights move between image pairs!

Repeatability vs number of scales sampled per octave



David G. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, 60, 2 (2004), pp. 91-110

Sensitivity to number of histogram orientations

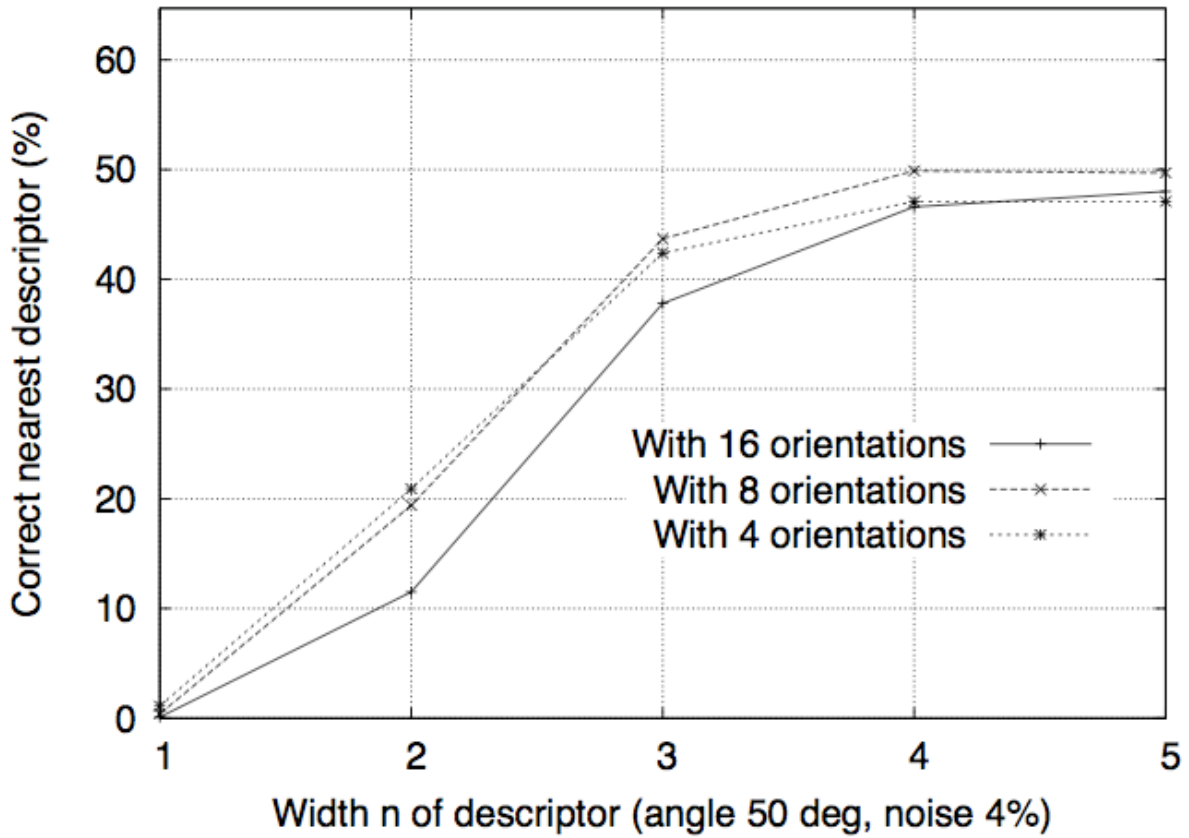
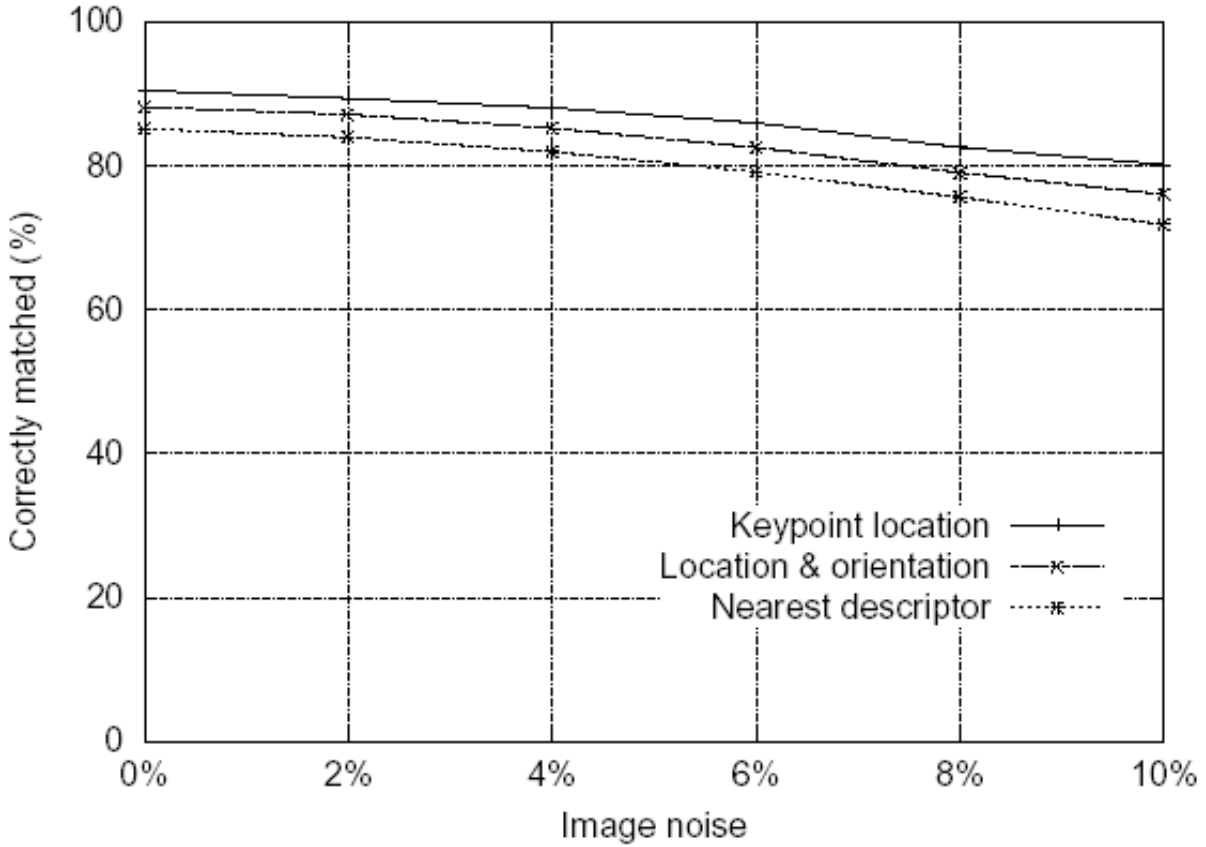


Figure 8: This graph shows the percent of keypoints giving the correct match to a database of 40,000 keypoints as a function of width of the $n \times n$ keypoint descriptor and the number of orientations in each histogram. The graph is computed for images with affine viewpoint change of 50 degrees and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

David G. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, 60, 2 (2004), pp. 91-110

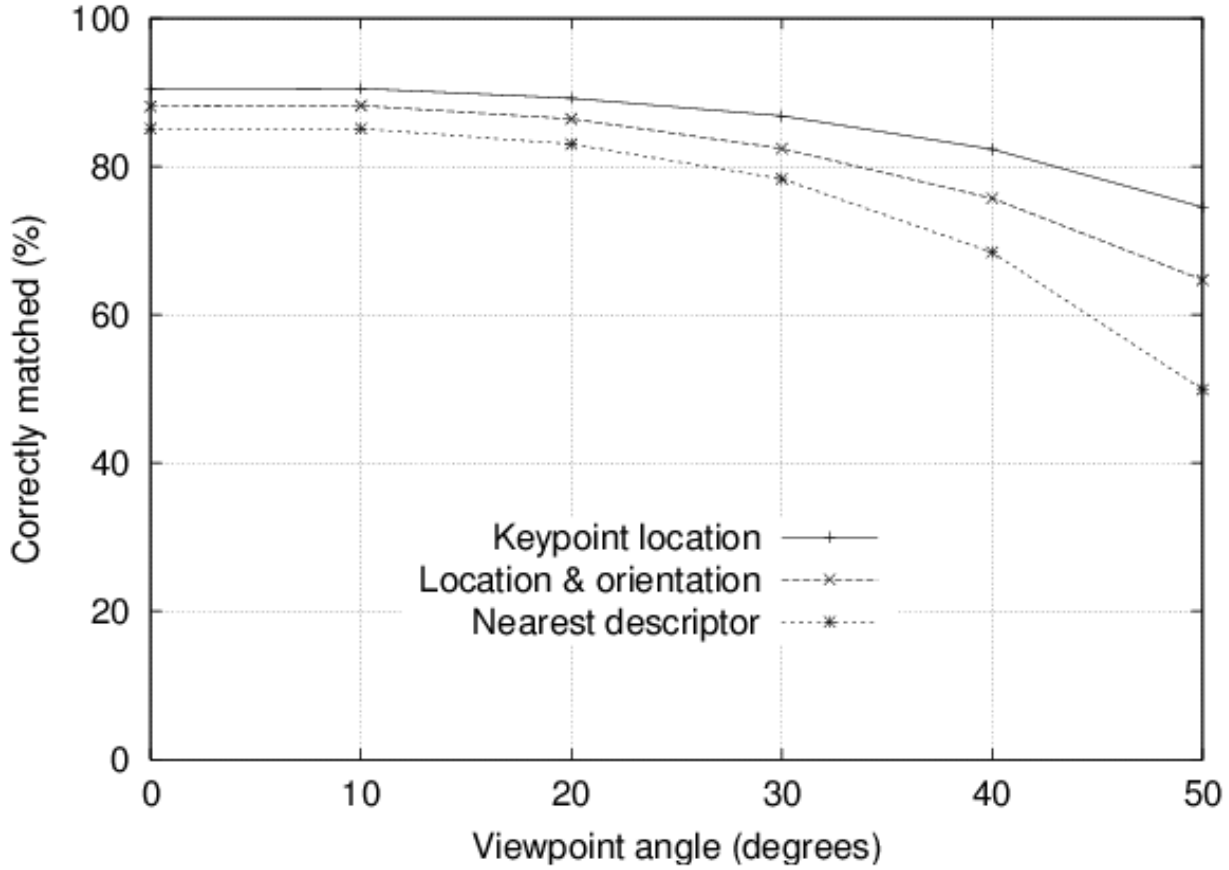
Feature stability to noise

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



Feature stability to affine change

- Match features after random change in image scale & orientation, with 2% image noise, and affine distortion
- Find nearest neighbor in database of 30,000 features



Distinctiveness of features

- Vary size of database of features, with 30 degree affine change, 2% image noise
- Measure % correct for single nearest neighbor match

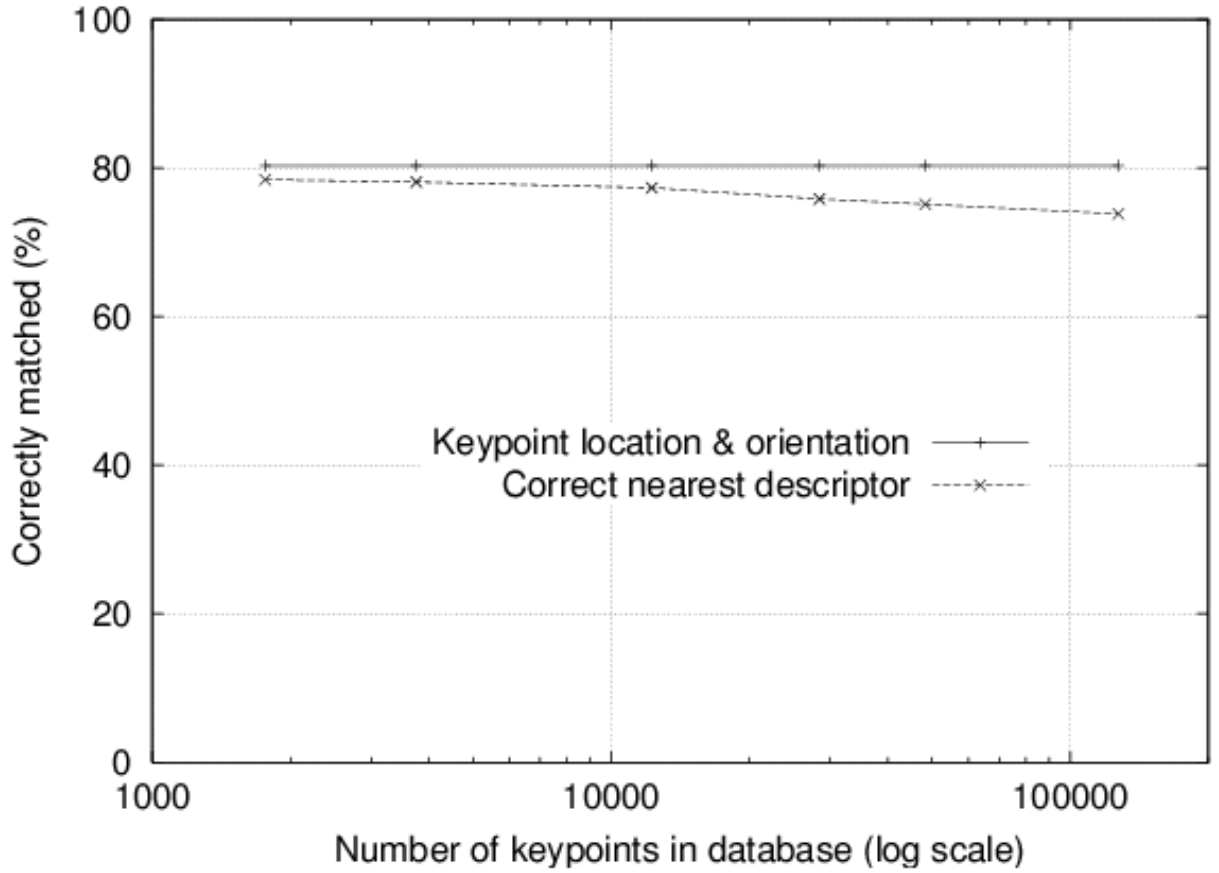


Figure 12: The training images for two objects are shown on the left. These can be recognized in a cluttered image with extensive occlusion, shown in the middle. The results of recognition are shown on the right. A parallelogram is drawn around each recognized object showing the boundaries of the original training image under the affi ne transformation solved for during recognition. Smaller squares indicate the keypoints that were used for recognition.

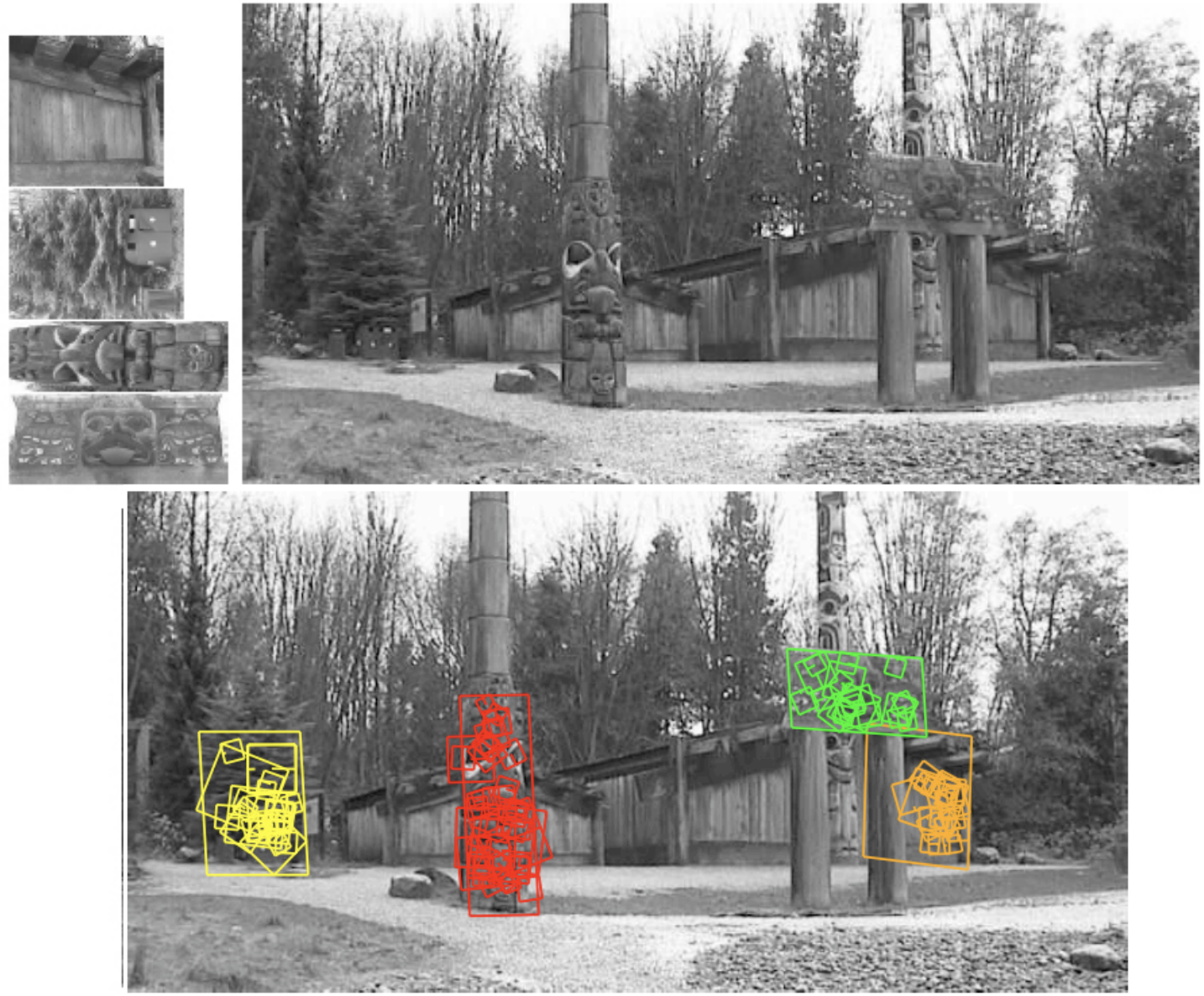


Figure 13: This example shows location recognition within a complex scene. The training images for locations are shown at the upper left and the 640x315 pixel test image taken from a different viewpoint is on the upper right. The recognized regions are shown on the lower image, with keypoints shown as squares and an outer parallelogram showing the boundaries of the training images under the affi ne transform used for recognition.

Nice SIFT resources

- an online tutorial:http://www.aishack.in/2010/05/sift-scale-invariant-feature-transform/
- Wikipedia:http://en.wikipedia.org/wiki/Scale-invariant_feature_transform

What we will learn today?

- SIFT: an image region descriptor
- HOG: another image descriptor
- Other image descriptors

- Matching
- Application: Panorama

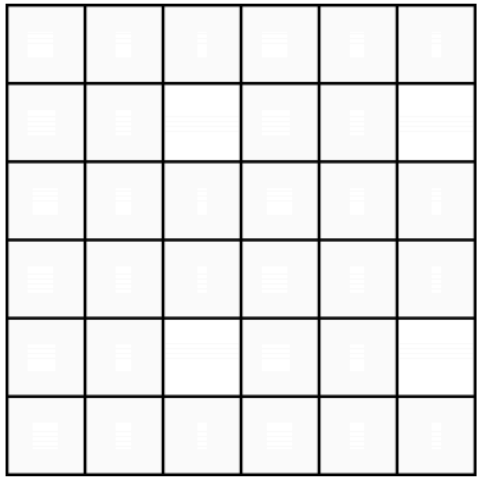
Histogram of Oriented Gradients

- Find robust feature set that allows object form to be discriminated.
- Challenges
  - Wide range of pose and large variations in appearances
  - Cluttered backgrounds under different illumination
  - “Speed” for mobile vision
- References
  - [1] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In CVPR, pages 886-893, 2005
  - [2] Chandrasekhar et al. CHoG: Compressed Histogram of Gradients - A low bit rate feature descriptor, CVPR 2009

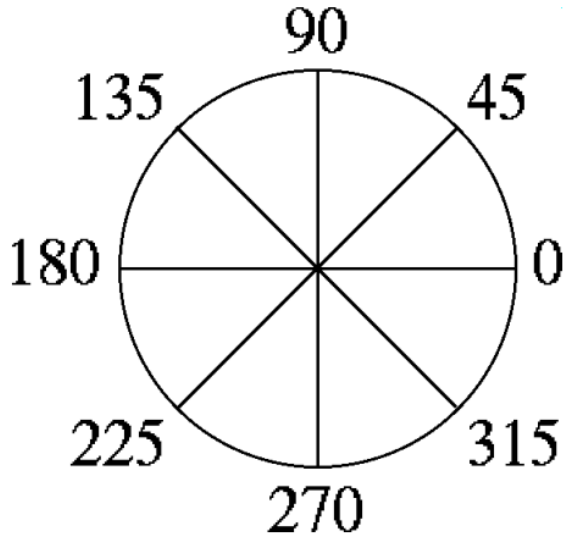
Histogram of Oriented Gradients

Local object appearance and shape can often be characterized rather well by the distribution of local intensity gradients or edge directions.

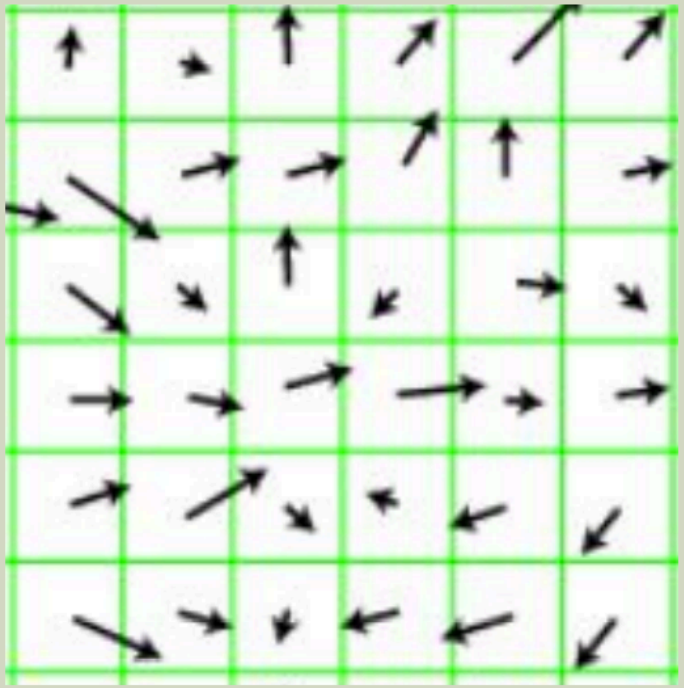
Histogram of Oriented Gradients

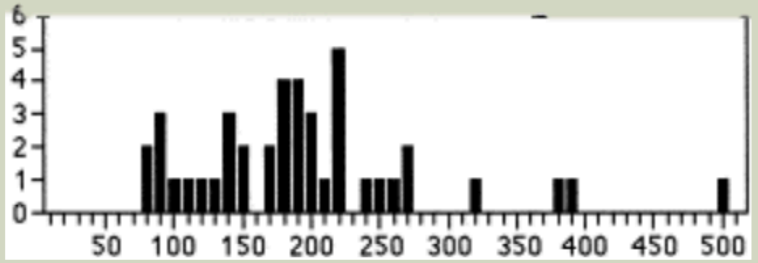


- Dividing the image window into small spatial regions (cells)
- Cells can be either rectangle or radial.
- Each cell accumulating a weighted local 1-D histogram of gradient directions over the pixels of the cell.

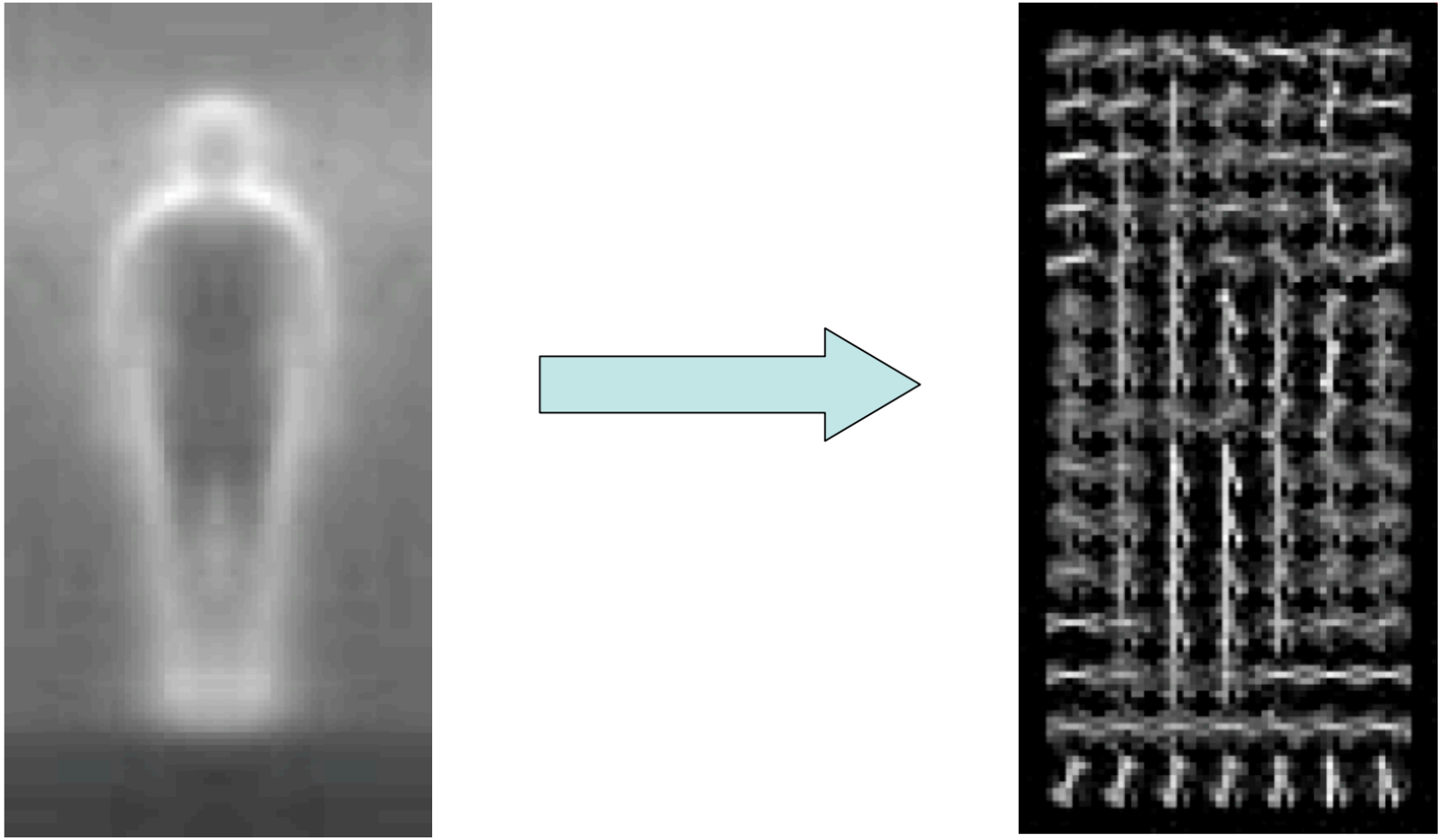


Histogram of Oriented Gradients



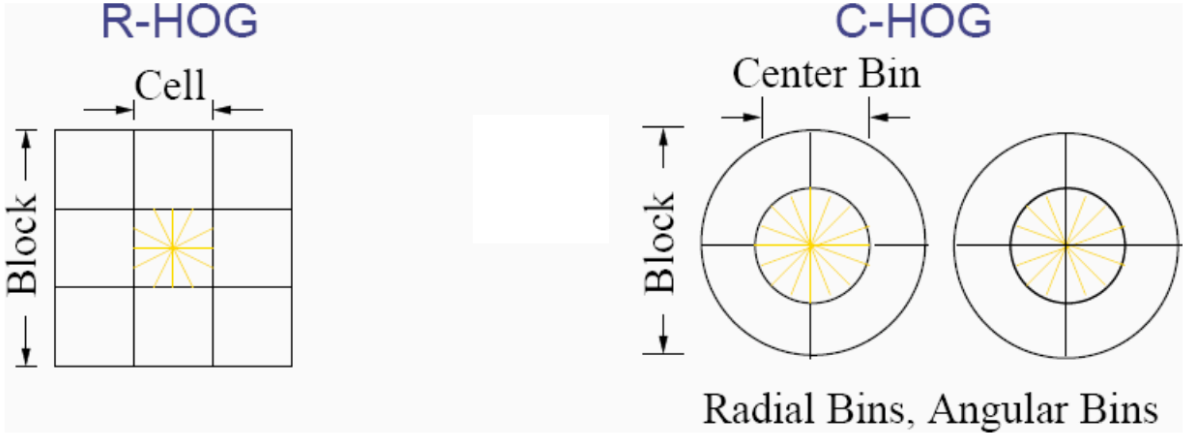


Histogram of Oriented Gradients

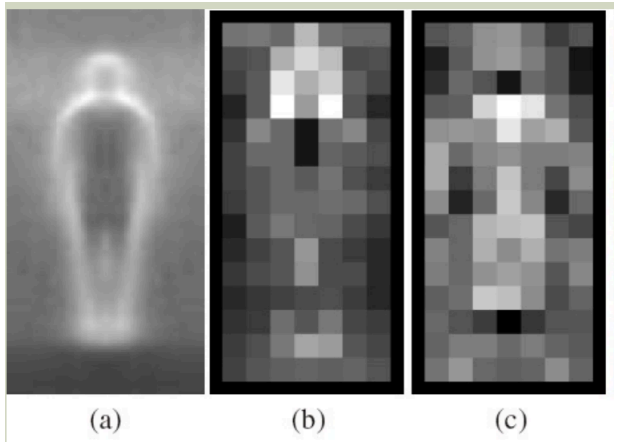


Normalization

- For better invariance to illumination and shadowing. it is useful to contrast-normalize the local responses before using them.
- Accumulate local histogram “energy” over a larger regions (“blocks”) to normalize all of the cells in the block.

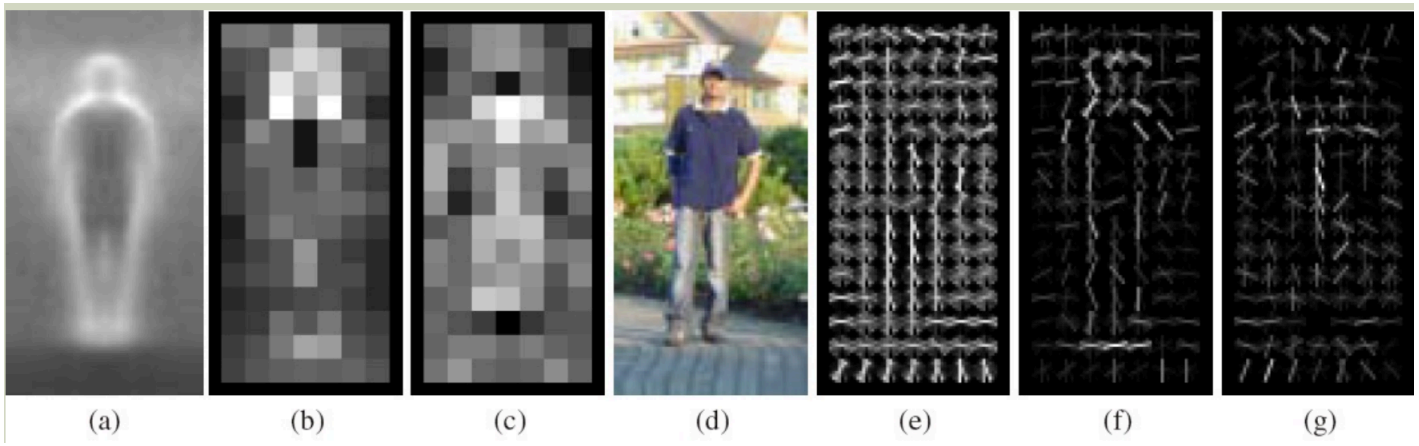


Visualizing HoG



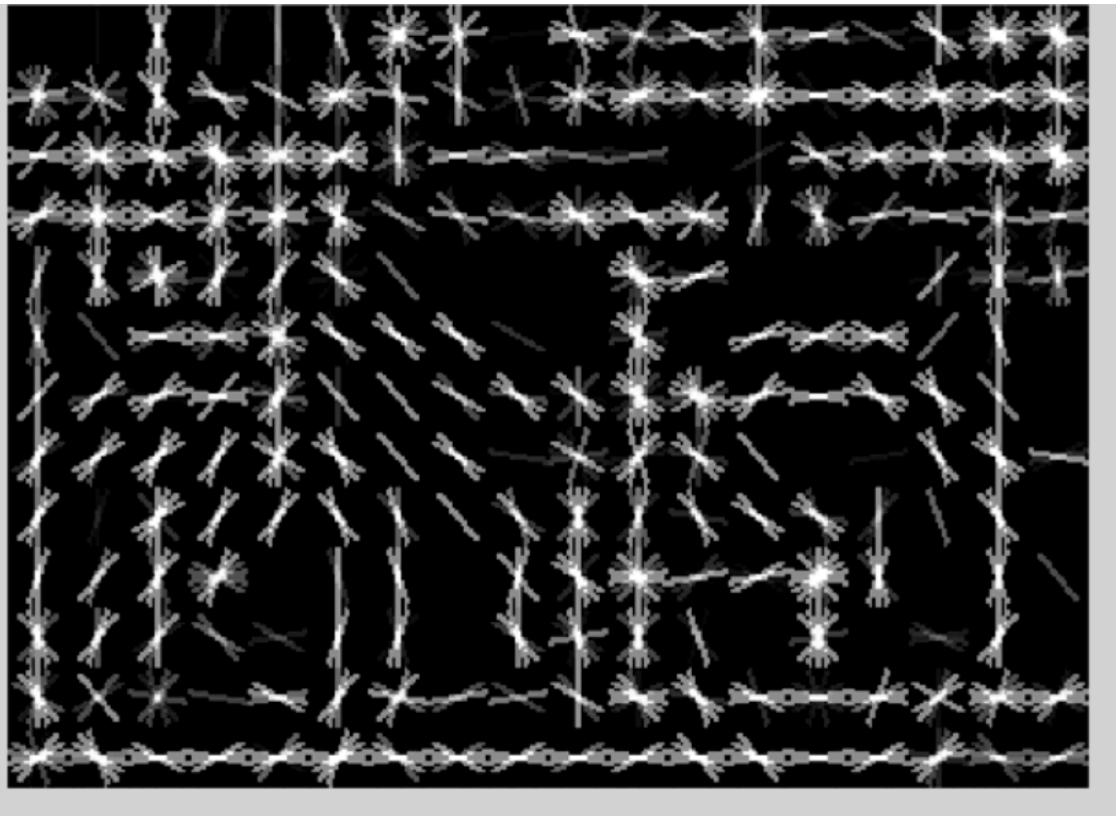
a. Average gradient over positive examples
b. Maximum positive weight in each block
c. Maximum negative weight in each block
d. A test image
e. It’s R-HOG descriptor
f. R-HOG descriptor weighted by positive weights
g. R-HOG descriptor weighted by negative weights

Visualizing HoG



a. Average gradient over positive examples
b. Maximum positive weight in each block
c. Maximum negative weight in each block
d. A test image
e. It’s R-HOG descriptor
f. R-HOG descriptor weighted by positive weights
g. R-HOG descriptor weighted by negative weights

Visualizing HoG



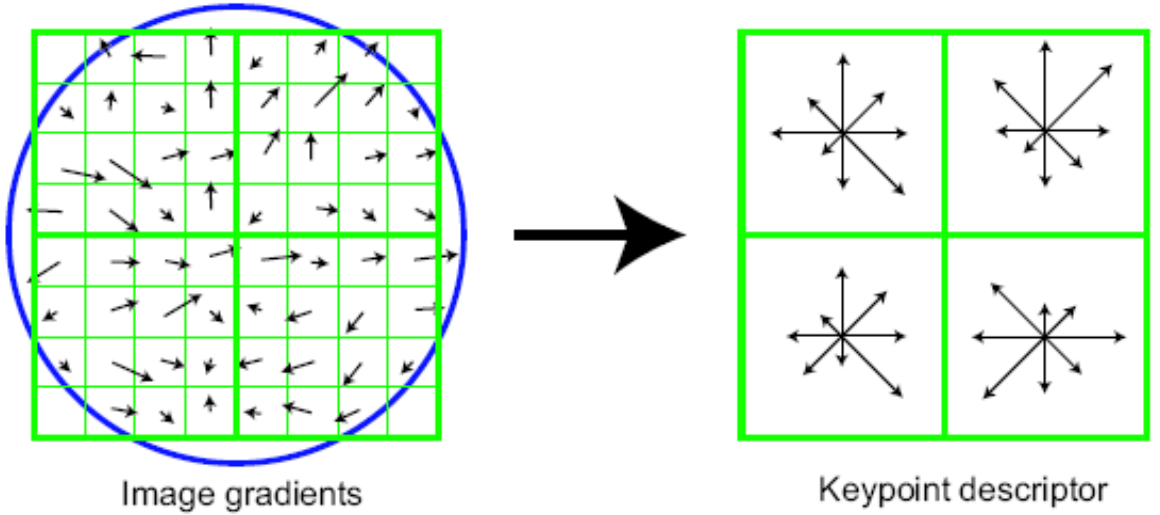
Difference between HoG and SIFT

- HoG is usually used to describe entire images. SIFT is used for key point matching
- SIFT histrograms are oriented towards the dominant gradient. HoG is not.
- HoG gradients are normalized using neighborhood bins.
- SIFT descriptors use varying scales to compute multiple descriptors.

Review: Local Descriptors

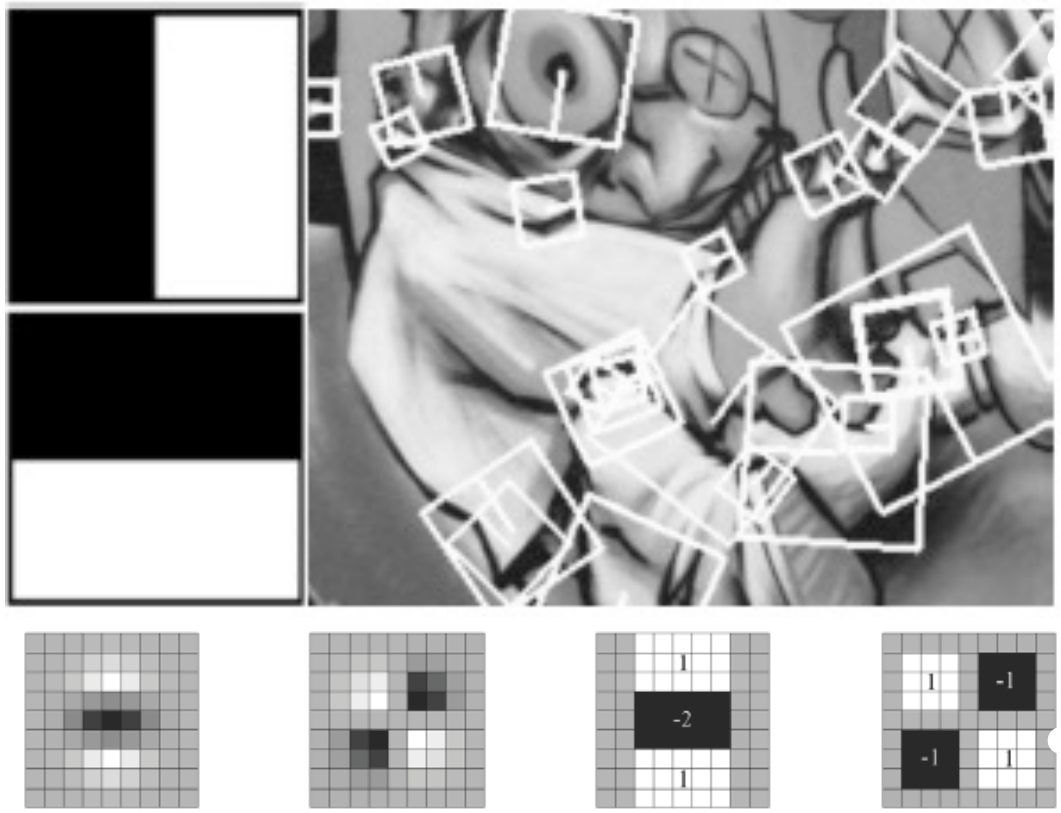
- Most features can be thought of as templates, histograms (counts), or combinations

- The ideal descriptor should be
  - Robust and Distinctive
  - Compact and Efficient



- Most available descriptors focus on edge/gradient information
  - Capture texture information
  - Color rarely used

Local Descriptors: SURF



Fast approximation of SIFT idea

Efficient computation by 2D box filters & integral images

⇒ 6 times faster than SIFT

Equivalent quality for object identification

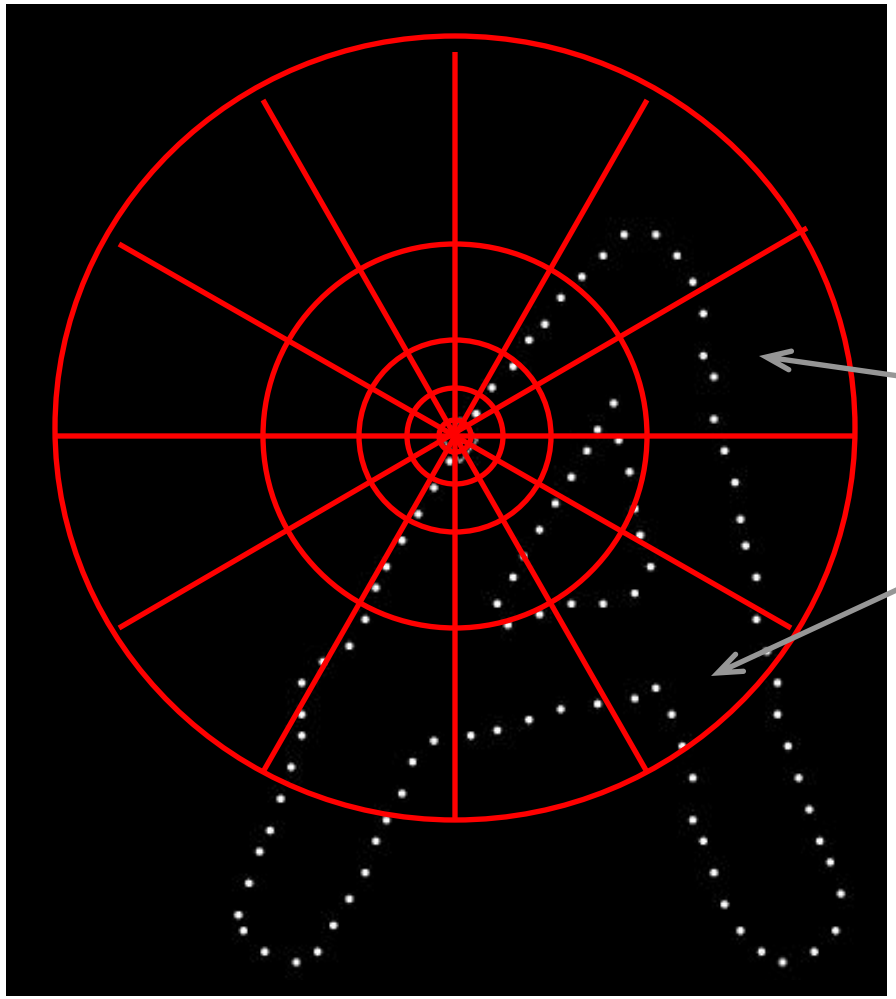
GPU implementation available

Feature extraction @ 200Hz
(detector + descriptor, 640×480 img)

http://www.vision.ee.ethz.ch/~surf

[Bay, ECCV'06], [Cornelis, CVGPU'08]

Local Descriptors: Shape Context



Count the number of points inside each bin, e.g.:

Count = 4

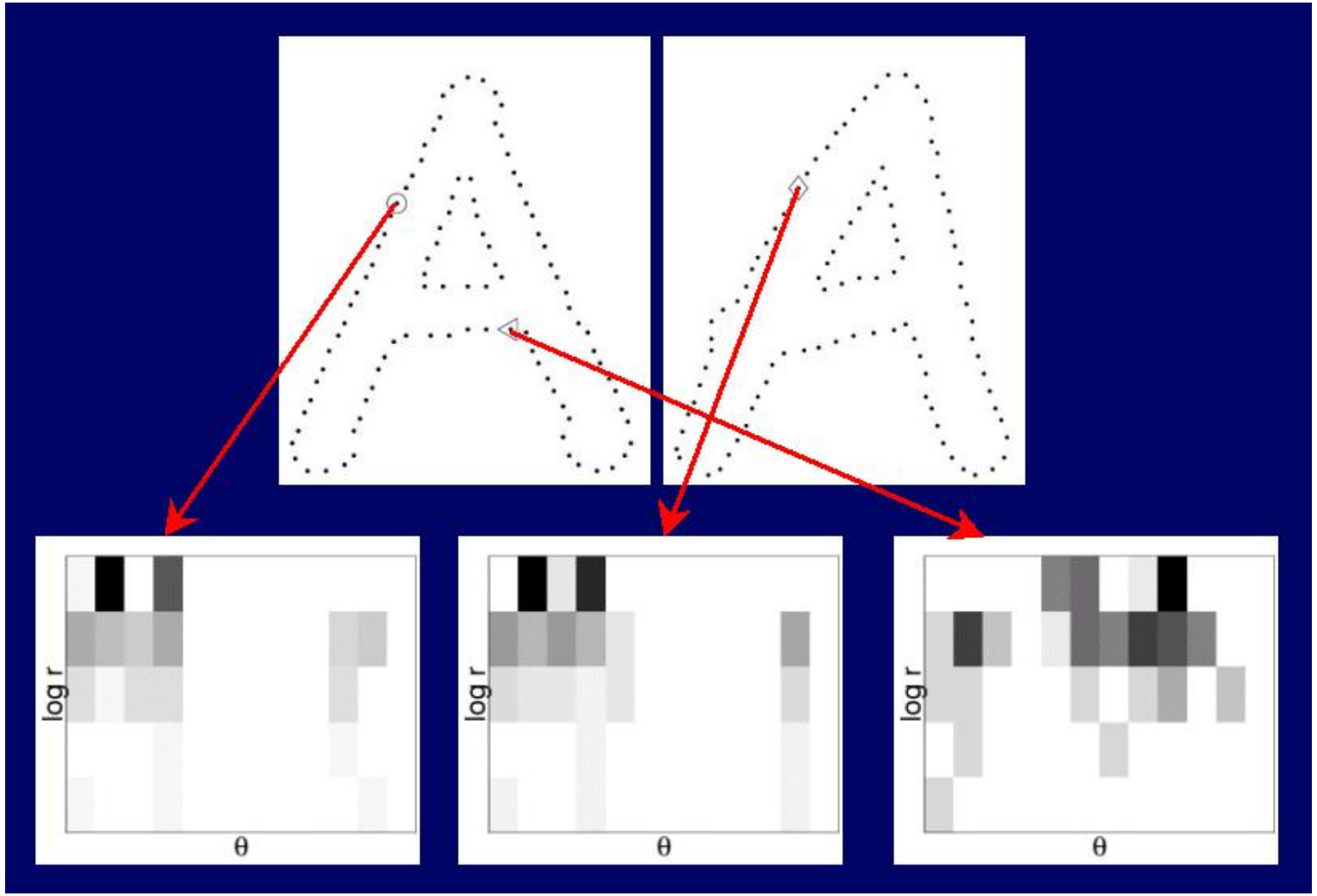
⋮

Count = 10

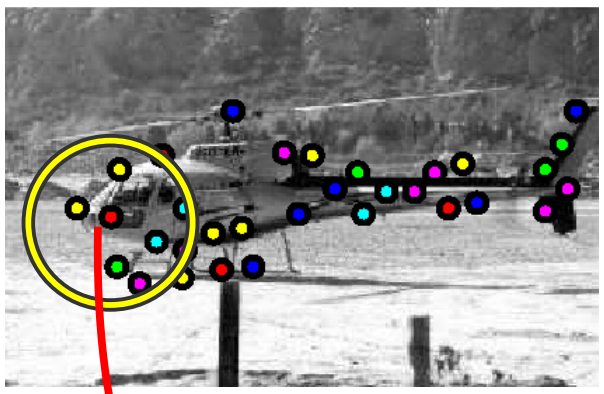
Log-polar binning:
More precision for nearby points, more flexibility for farther points.

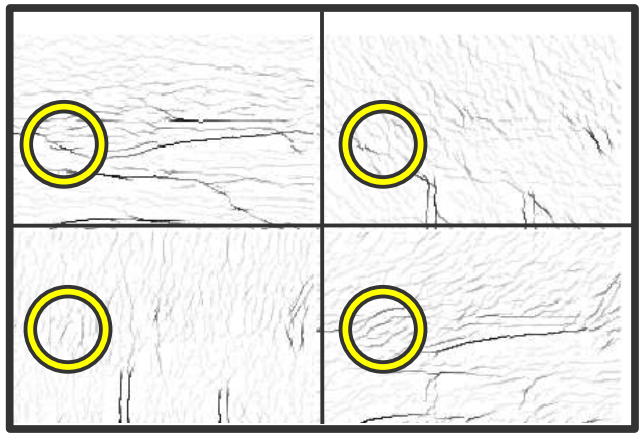
Belongie & Malik, ICCV 2001

Shape Context Descriptor



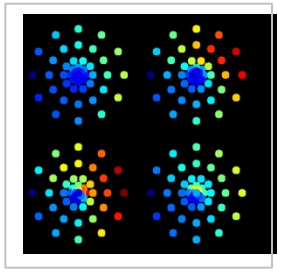
Local Descriptors: Geometric Blur



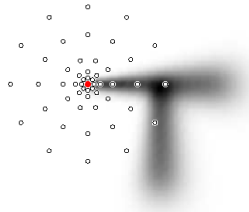


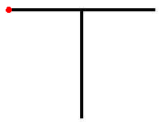
Compute edges at four orientations

Extract a patch in each channel



Example descriptor





Apply spatially varying blur and sub-sample

(Idealized signal)

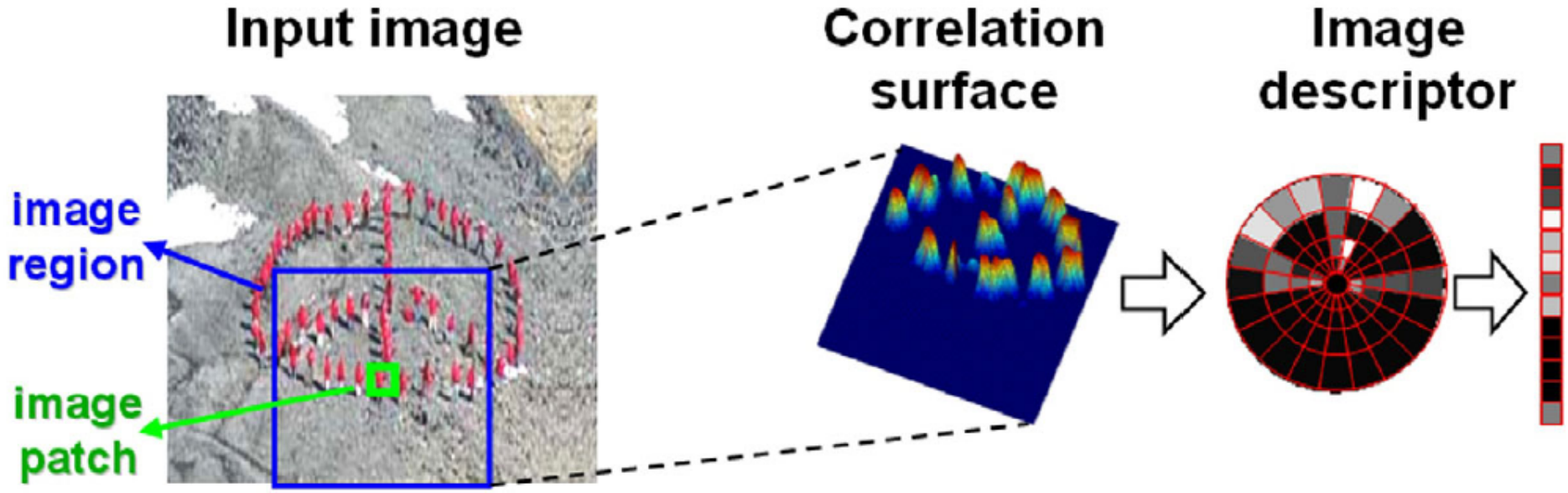
Berg & Malik, CVPR 2001

Self-similarity Descriptor

Figure 1. *These images of the same object (a heart) do NOT share common image properties (colors, textures, edges), but DO share a similar geometric layout of local internal self-similarities.*

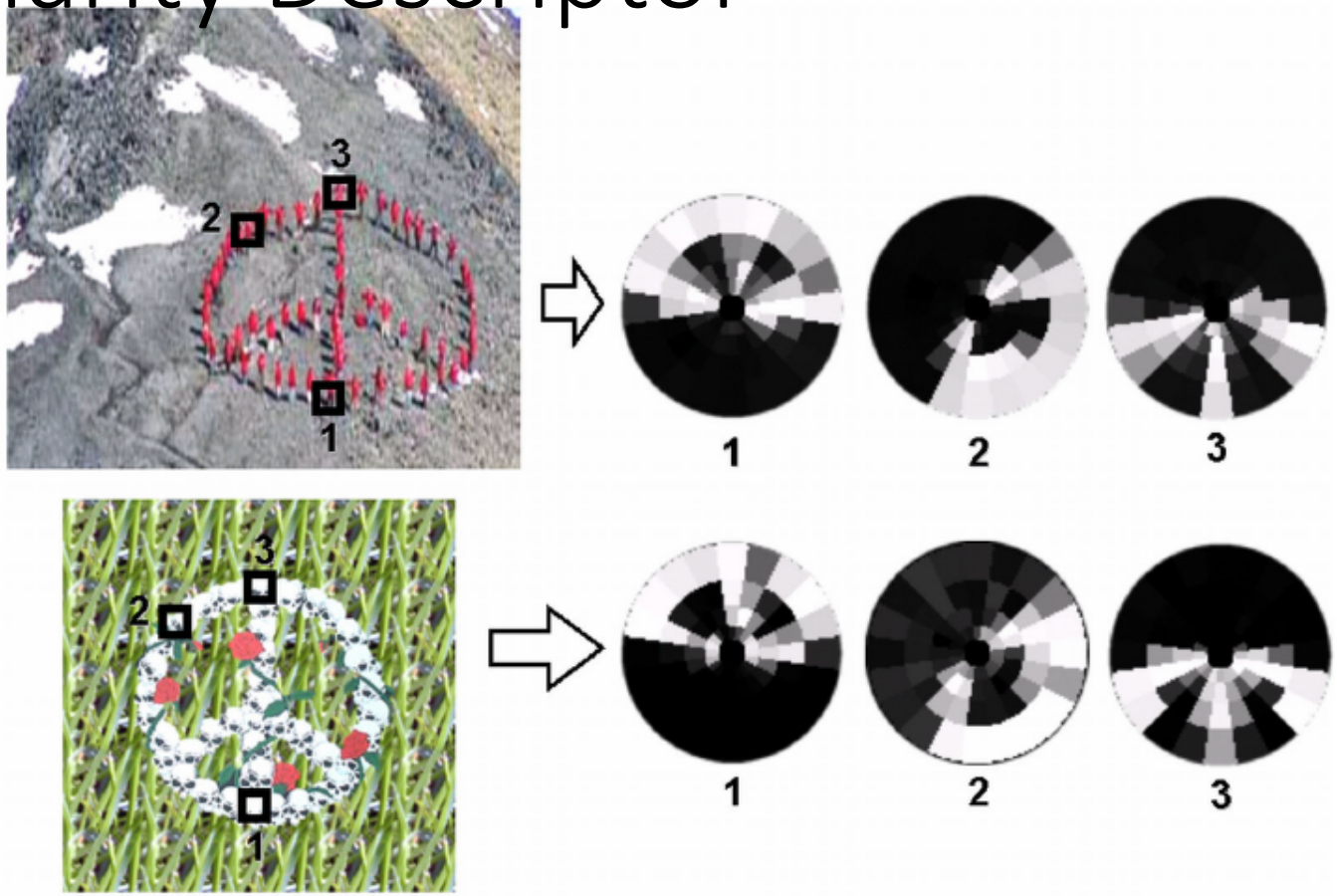
Matching Local Self-Similarities across Images and Videos, Shechtman and Irani, 2007

Self-similarity Descriptor



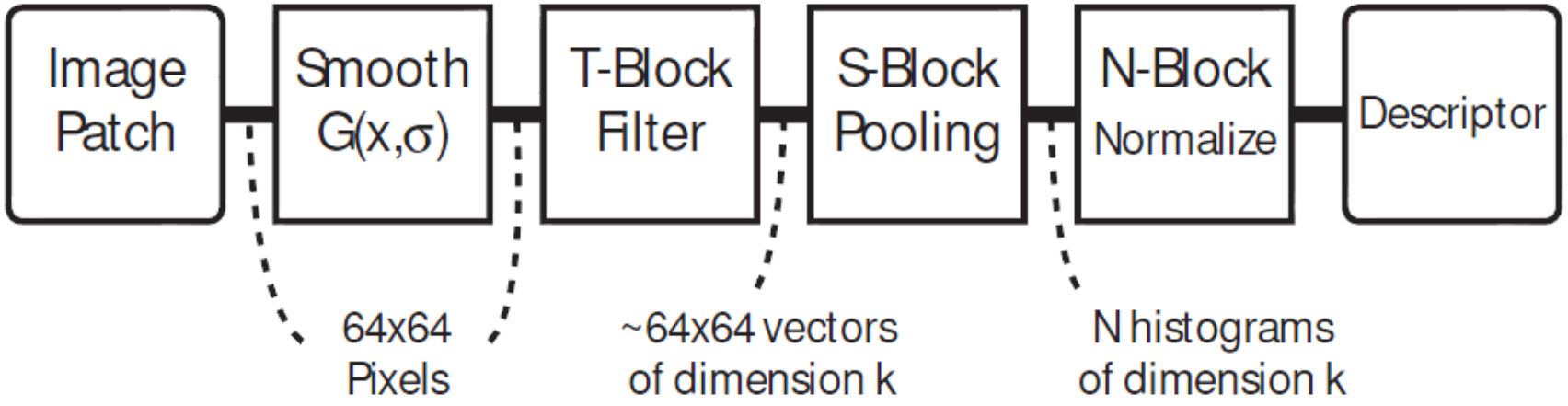
Matching Local Self-Similarities across Images and Videos, Shechtman and Irani, 2007

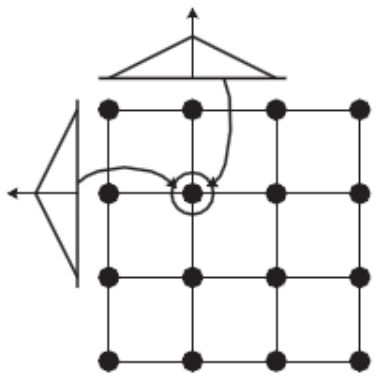
Self-similarity Descriptor



Matching Local Self-Similarities across Images and Videos, Shechtman and Irani, 2007

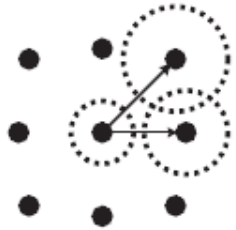
Learning Local Image Descriptors
Winder and Brown, 2007



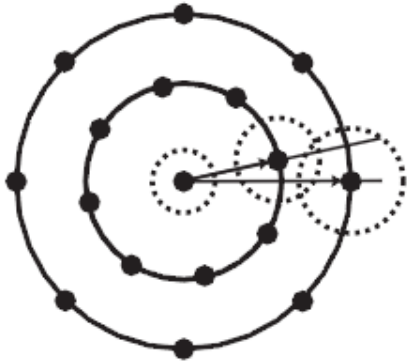


S1: SIFT grid with bilinear weights

S2: GLOH polar grid with bilinear radial and angular weights



S3: 3x3 grid with Gaussian weights



S4: 17 polar samples with Gaussian weights

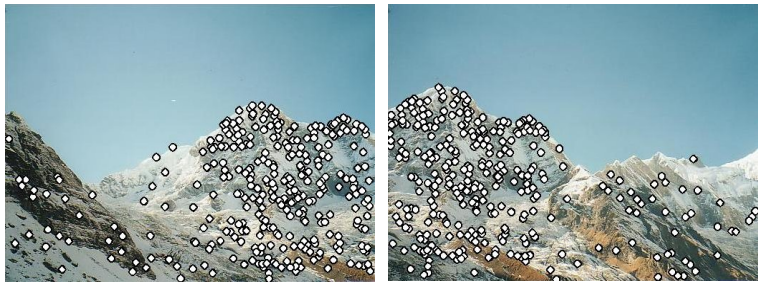
What we will learn today?

- SIFT: an image region descriptor
- HOG: another image descriptor
- Other image descriptors

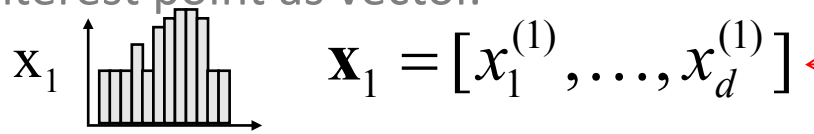
- Matching [Read Szeliski 4.1]
- Application: Panorama

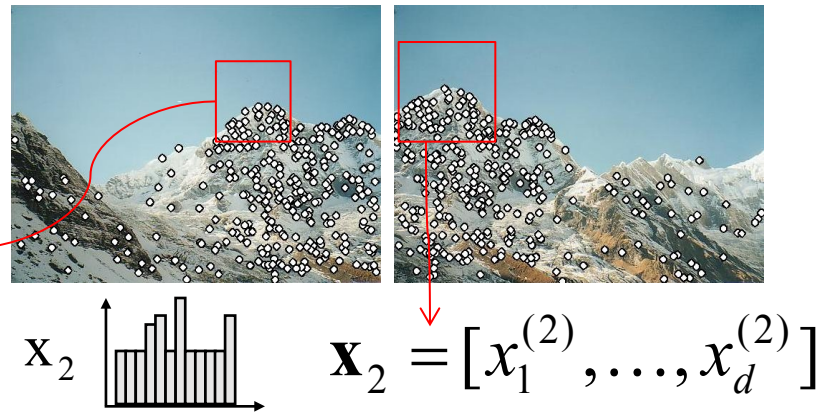
Local features: main components

1) Detection:
Find a set of distinctive key points.

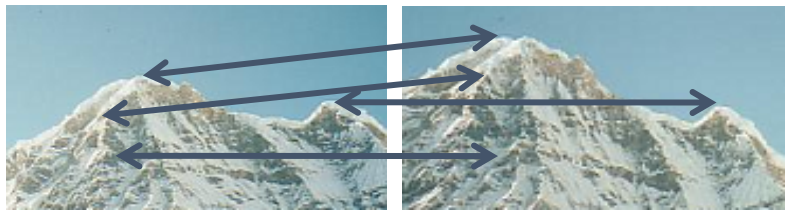


2) Description:
Extract feature descriptor around each interest point as vector.

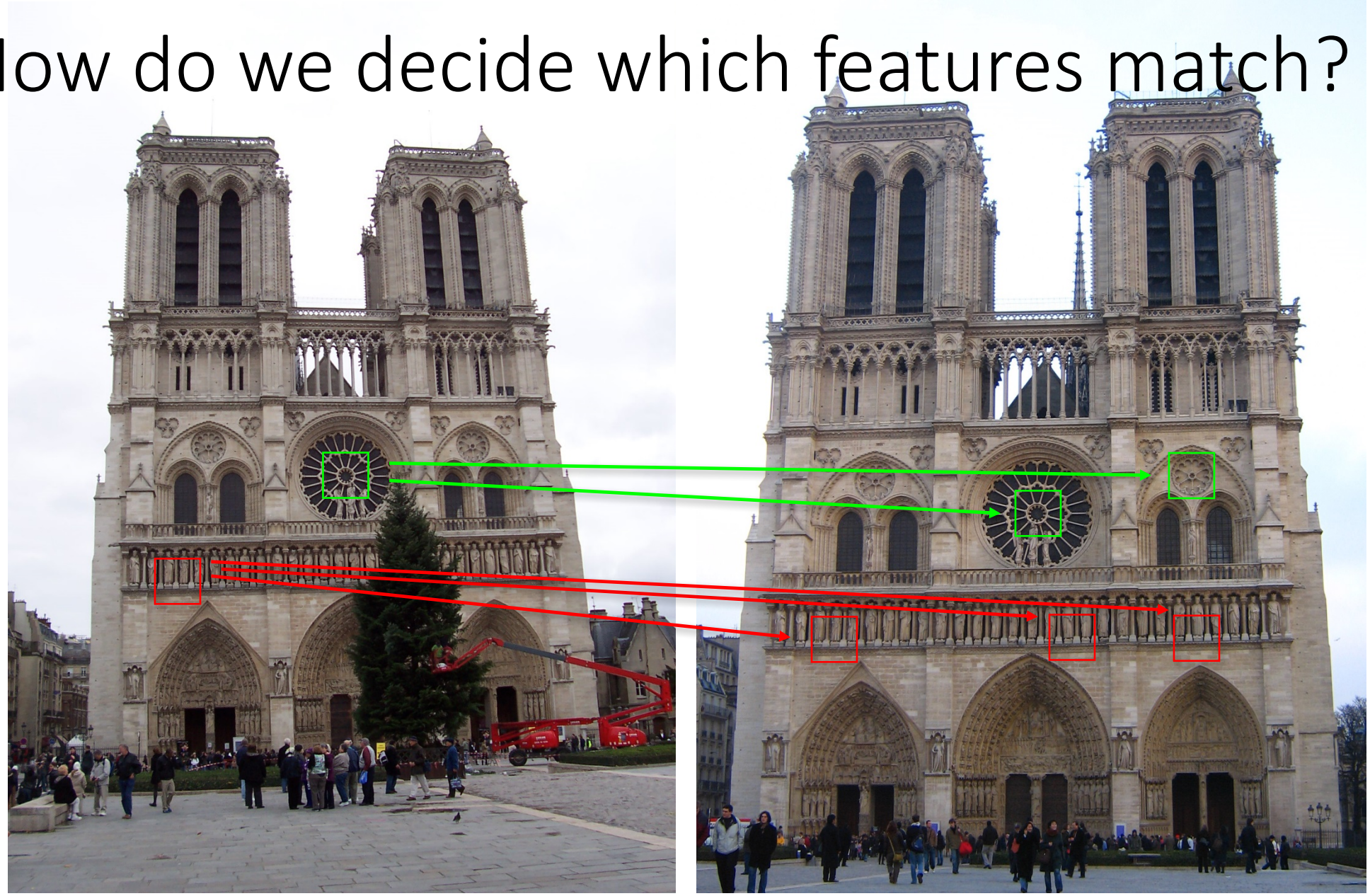




3) Matching:
Compute distance between feature vectors to find correspondence.



How do we decide which features match?

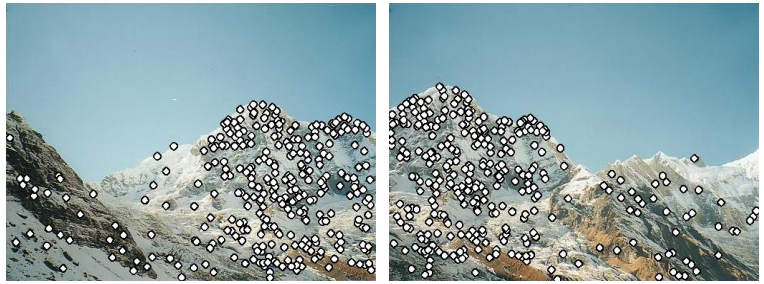


Distance: 0.34, 0.30, 0.40

Distance: 0.61, 1.22

Think-Pair-Share

- *Design a feature point matching scheme.*
- Two images, I_1 and I_2



- Two sets X_1 and X_2 of feature points
  - Each feature point $\mathbf{x_1}$ has a descriptor

$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$

- Distance, bijective/injective/surjective, noise, confidence, computational complexity, generality…

Euclidean distance vs. Cosine Similarity

- Euclidean distance:

$$\mathrm{d}(\mathbf{p}, \mathbf{q}) = \mathrm{d}(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

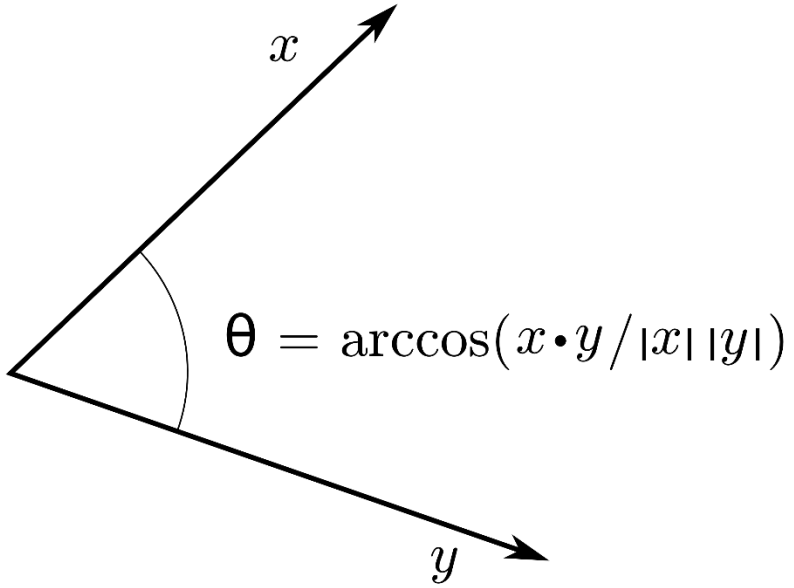
$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

$$\|\mathbf{q} - \mathbf{p}\| = \sqrt{(\mathbf{q} - \mathbf{p}) \cdot (\mathbf{q} - \mathbf{p})}.$$

- Cosine similarity:

$$\mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\|_2 \|\mathbf{b}\|_2 \cos\theta$$

$$\mathbf{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|_2 \|\mathbf{B}\|_2}$$



Wikipedia

Feature Matching

- Criteria 1:
  - Compute distance in feature space, e.g., Euclidean distance between 128-dim SIFT descriptors
  - Match point to lowest distance (nearest neighbor)

- Problems:
  - Does everything have a match?

Feature Matching

- Criteria 2:
  - Compute distance in feature space, e.g., Euclidean distance between 128-dim SIFT descriptors
  - Match point to lowest distance (nearest neighbor)
  - Ignore anything higher than threshold (no match!)

- Problems:
  - Threshold is hard to pick
  - Non-distinctive features could have lots of close matches, only one of which is correct

Nearest Neighbor Distance Ratio

Compare distance of closest (NN1) and second-closest (NN2) feature vector neighbor.

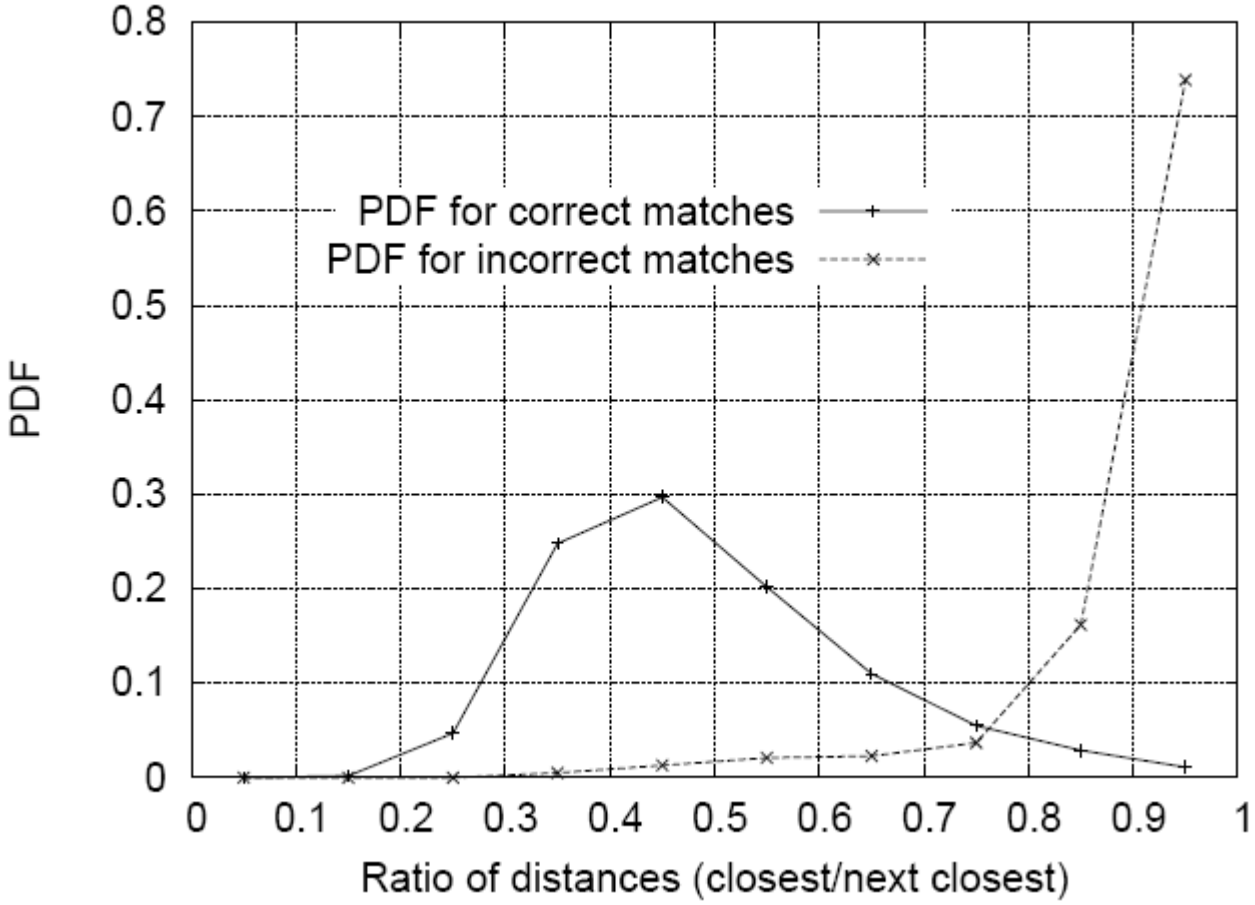
- If NN1 ≈ NN2, ratio $\frac{NN1}{NN2}$ will be ≈ 1 -> matches too close.
- As NN1 << NN2, ratio $\frac{NN1}{NN2}$ tends to 0.

Sorting by this ratio puts matches in order of confidence.

Threshold ratio – but how to choose?

Nearest Neighbor Distance Ratio

- Lowe computed a probability distribution functions of ratios
- 40,000 keypoints with hand-labeled ground truth



Ratio threshold depends on your application's view on the trade-off between the number of false positives and true positives!

Lowe IJCV 2004

Efficient compute cost

- Naïve looping: Expensive

- Operate on matrices of descriptors
- E.g., for row vectors,

  $\texttt{features\_image1} * \texttt{features\_image2}^{T}$

  produces matrix of dot product results for all pairs of features (cosine similarity).
  What can we do for Euclidean distance?

Live SIFT Demo

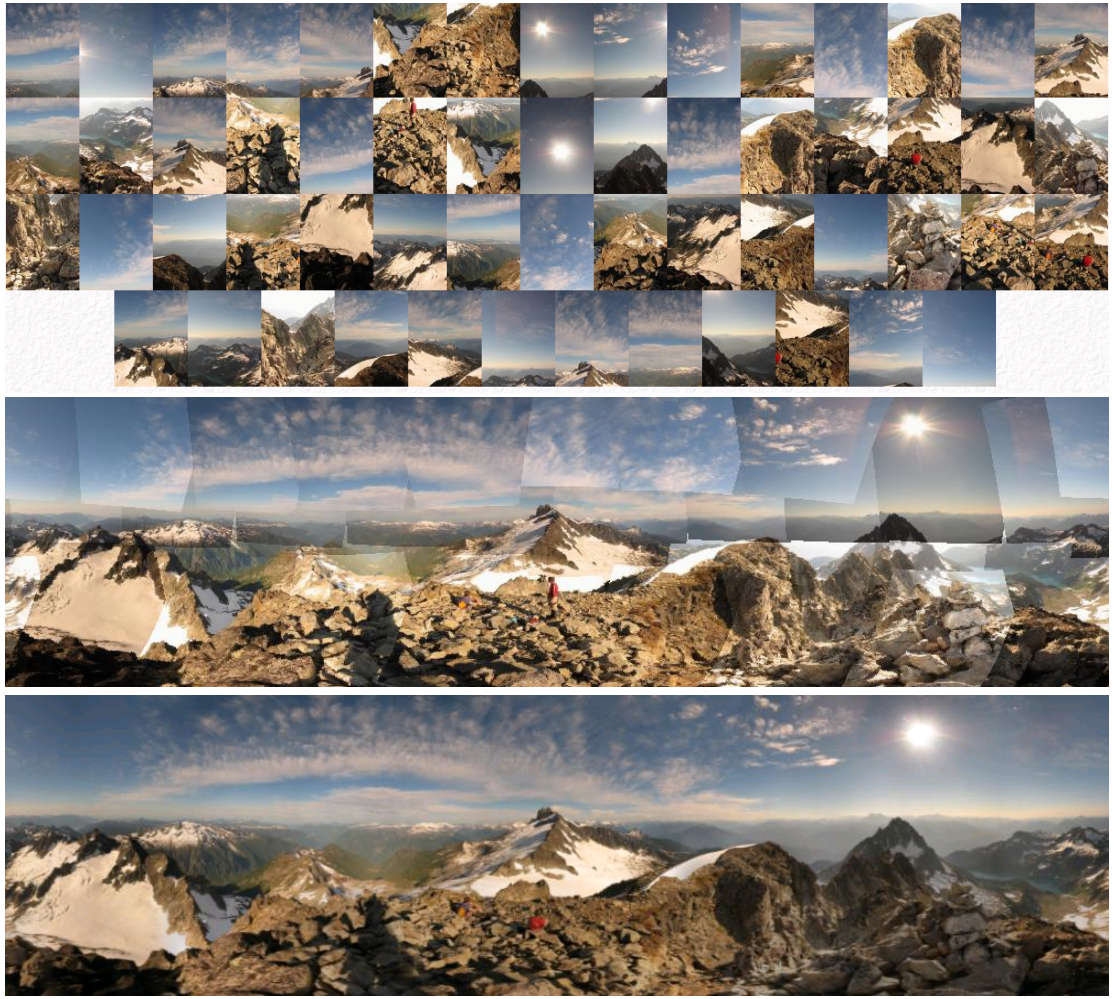
Actually a similar alternative, called SURF.

Speeded-Up Robust Features

Bay et al., ECCV 2006

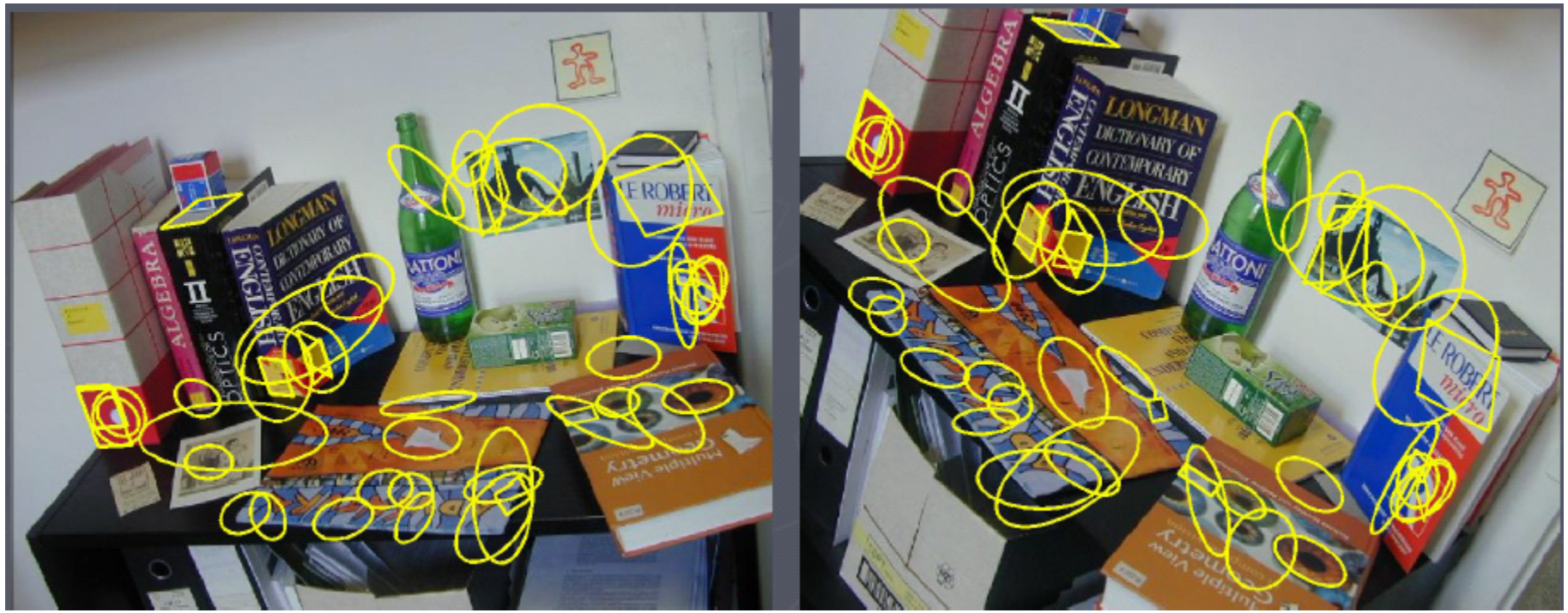
(Both are patented; SURF has non-commercial license.)

Automatic mosaicing



http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html

Wide baseline stereo



[Image from T. Tuytelaars ECCV 2006 tutorial]

Recognition of specific objects, scenes

Schmid and Mohr 1997

Sivic and Zisserman, 2003

Rothganger et al. 2003

Lowe 2002

Applications of local invariant features

- Wide baseline stereo
- Motion tracking
- **Panoramas**
- Mobile robot navigation
- 3D reconstruction
- Recognition
- …

What we will learn today?

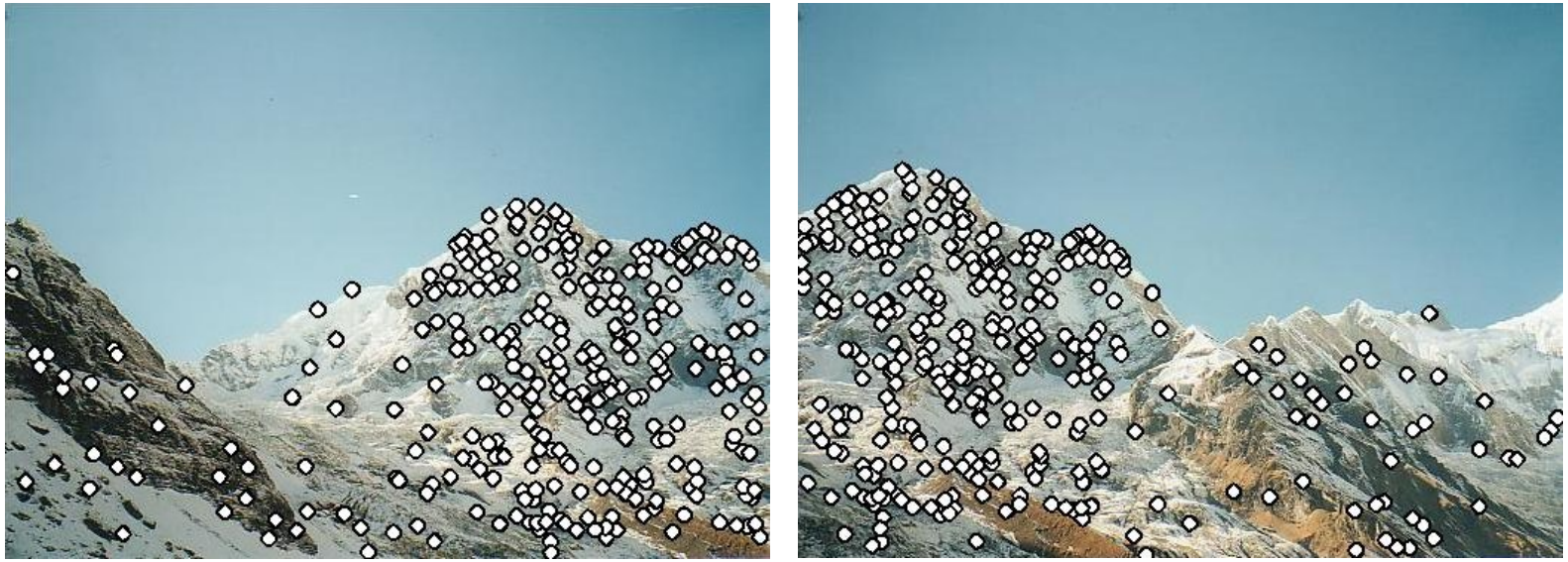
- SIFT: an image region descriptor
- HOG: another image descriptor
- Other image descriptors

- Matching [Read Szeliski 4.1]
- Application: Panorama

Application: Image Stitching

Slide credit: Darya Frolova, Denis Simakov

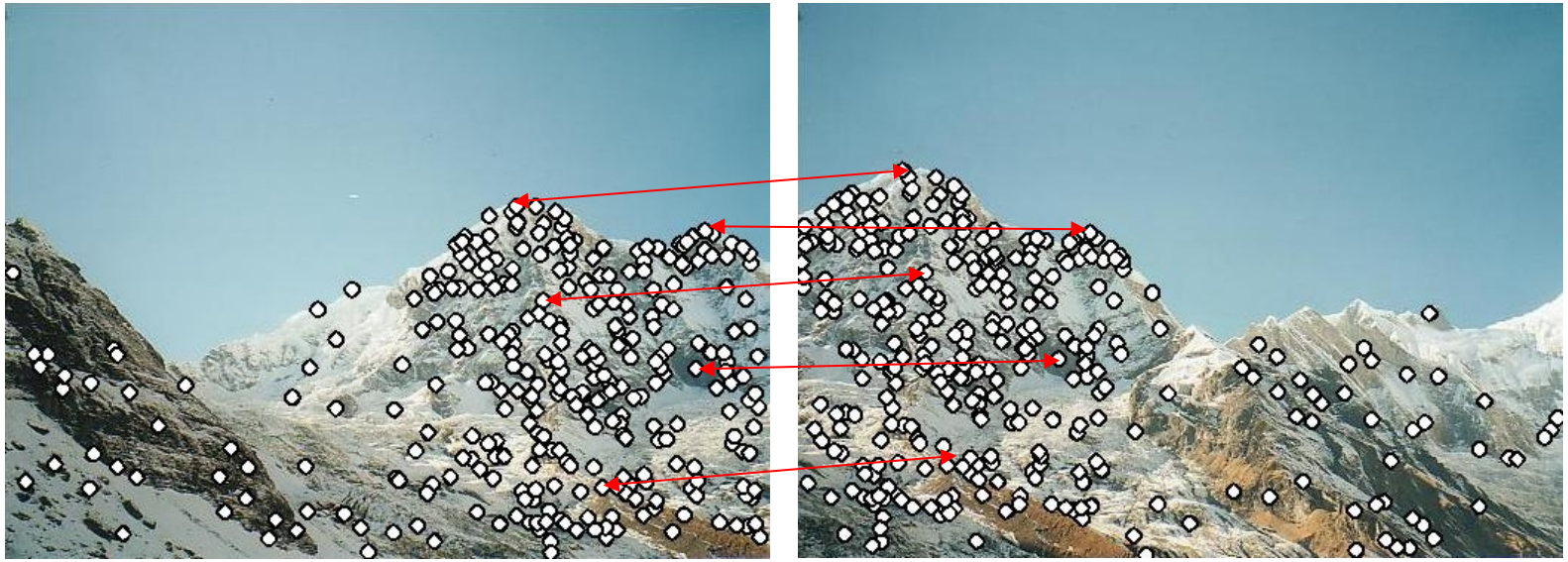
Application: Image Stitching



- Procedure:
  - Detect feature points in both images

Slide credit: Darya Frolova, Denis Simakov

Application: Image Stitching



- Procedure:
  - Detect feature points in both images
  - Find corresponding pairs

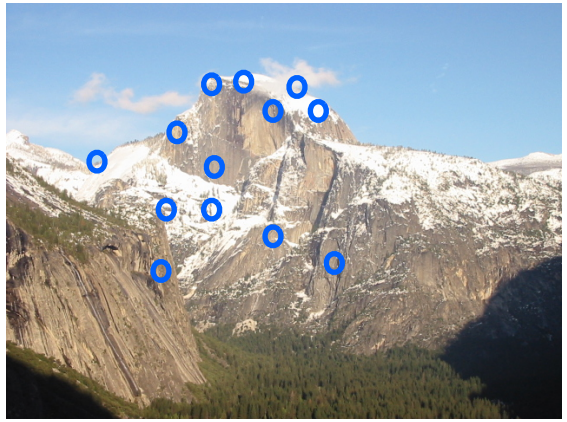
Slide credit: Darya Frolova, Denis Simakov

Application: Image Stitching

- Procedure:
  - Detect feature points in both images
  - Find corresponding pairs
  - Use these pairs to align the images

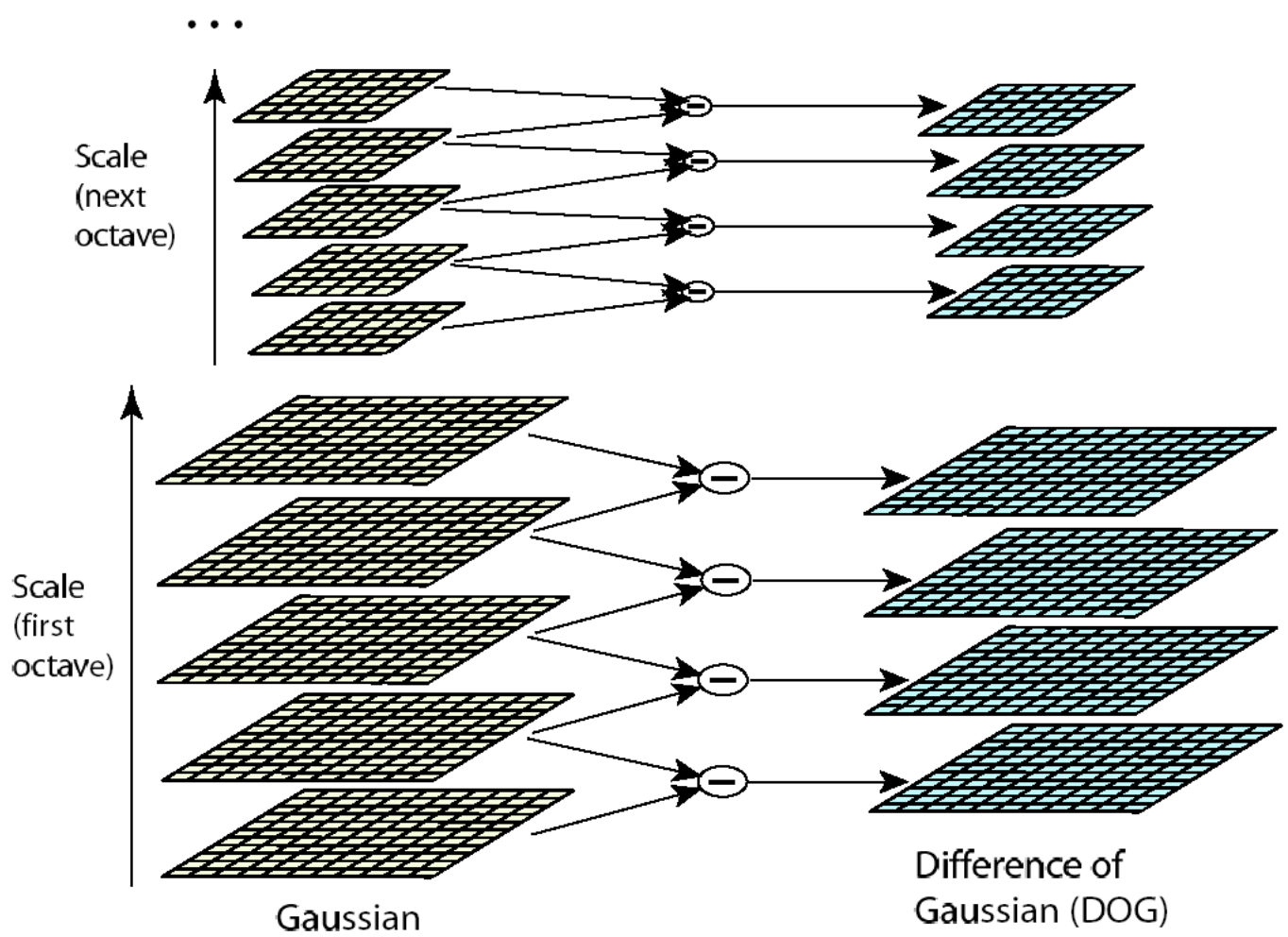
Slide credit: Darya Frolova, Denis Simakov

Main Flow

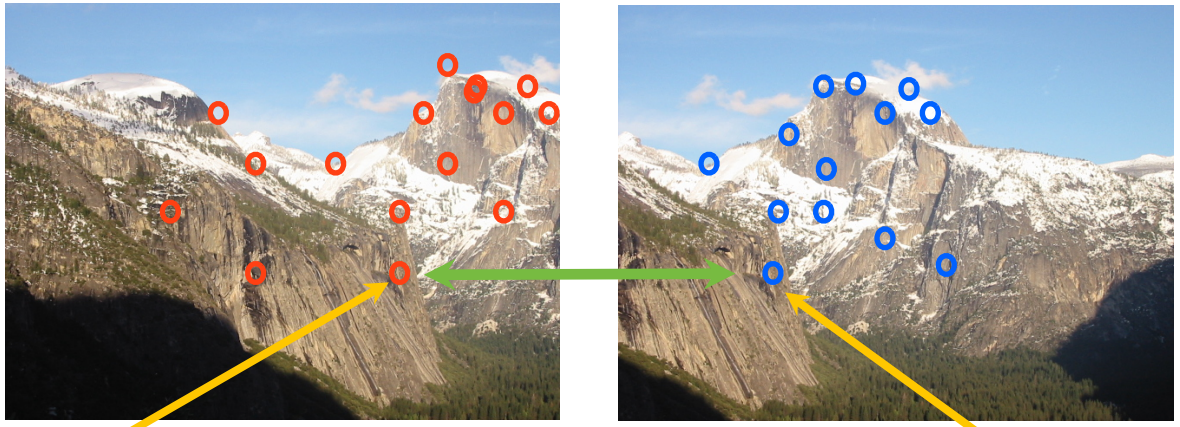


- Detect key points

Detect Key Points



Main Flow



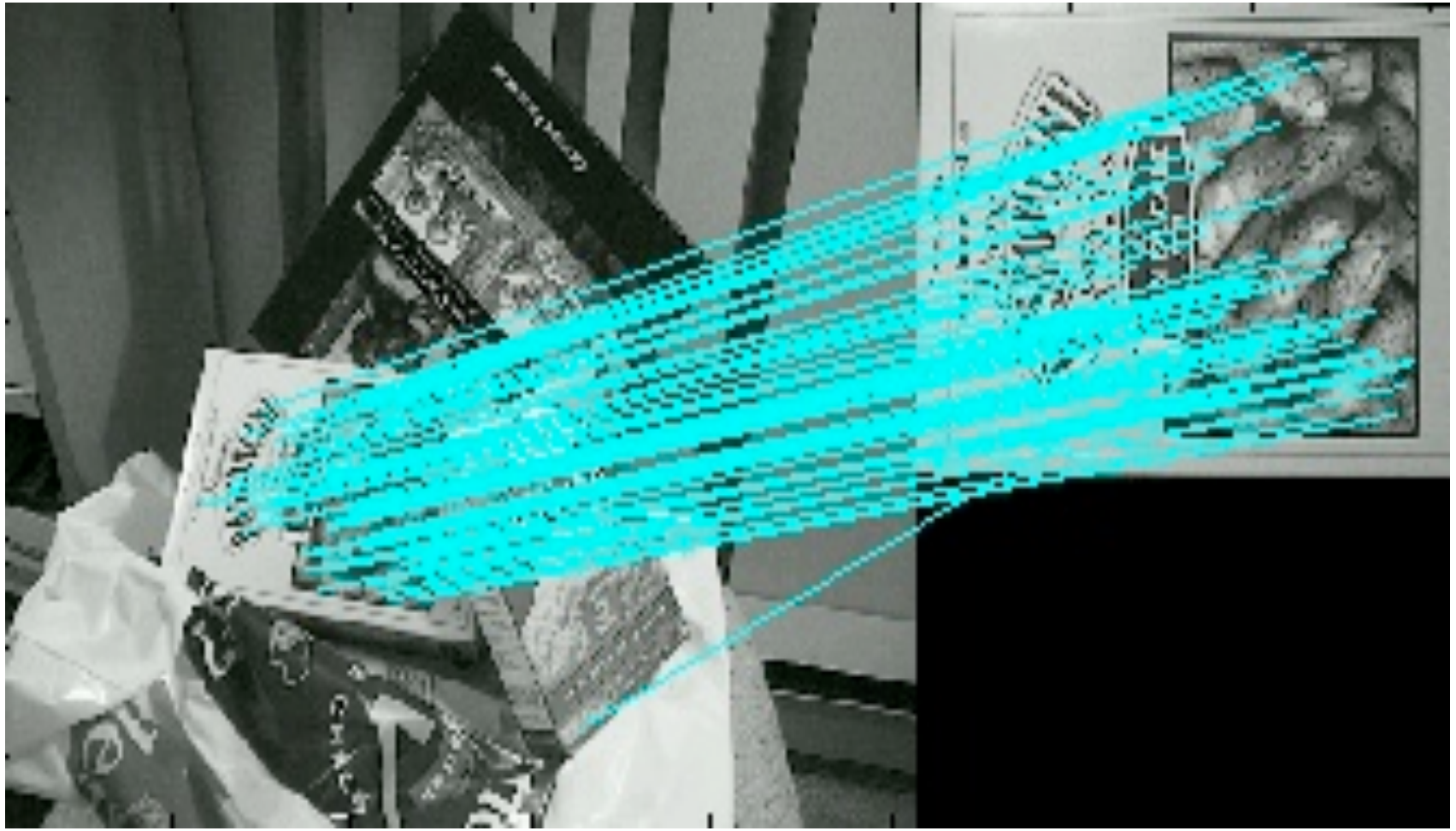
$(u_1, u_2, \ldots, u_{128})$

$(v_1, v_2, \ldots, v_{128})$

- Detect key points
- Build the SIFT descriptors
- Match SIFT descriptors

Match SIFT Descriptors

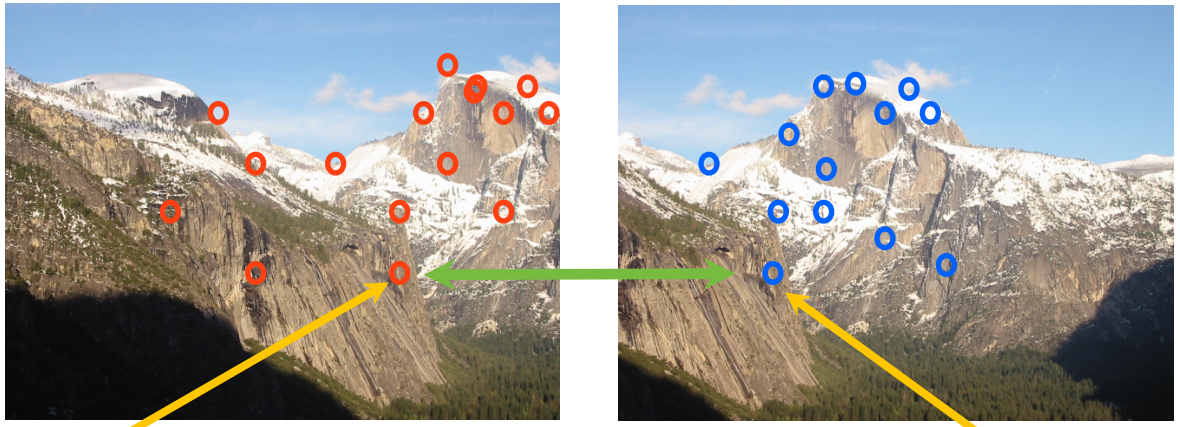
- Euclidean distance between descriptors



Skeleton Code

- Match SIFT descriptors (6 lines of code)
  - Input: D1, D2, thresh (default 0.7)
  - Output: match [D1's index, D2's index]
  - Try to use *one for* loop

Main Flow



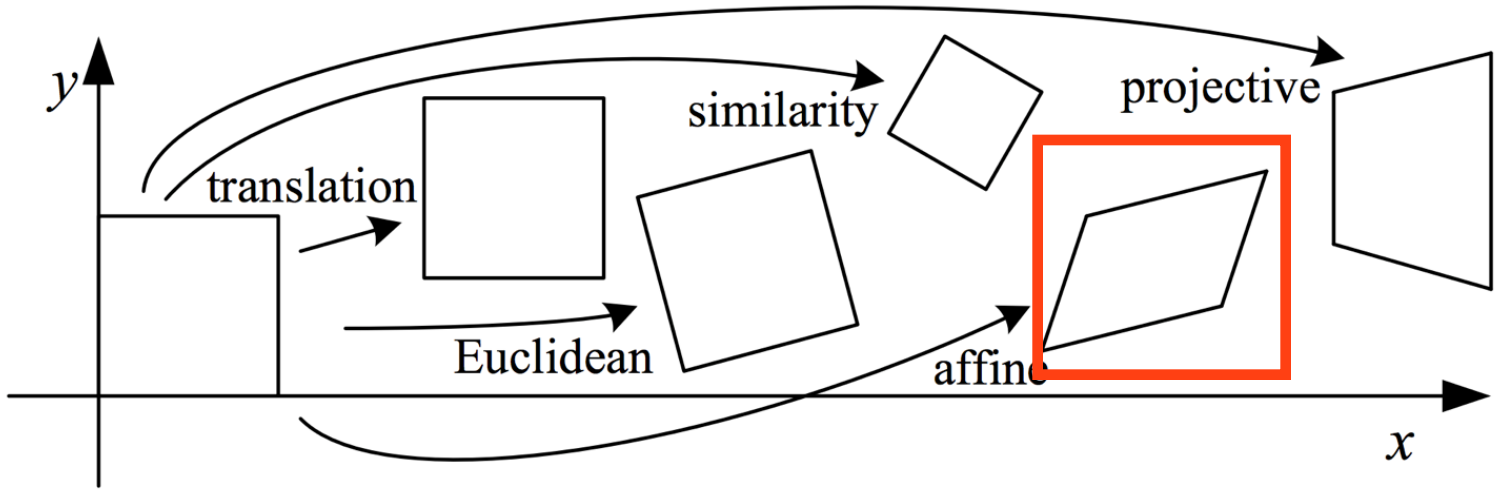
$(u_1, u_2, \ldots, u_{128})$ $(v_1, v_2, \ldots, v_{128})$

- Detect key points
- Build the SIFT descriptors
- Match SIFT descriptors
- Fitting the transformation

$$T = \begin{bmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ 0 & 0 & 1 \end{bmatrix}$$

Fitting the transformation

- 2D transformations



Skeleton Code

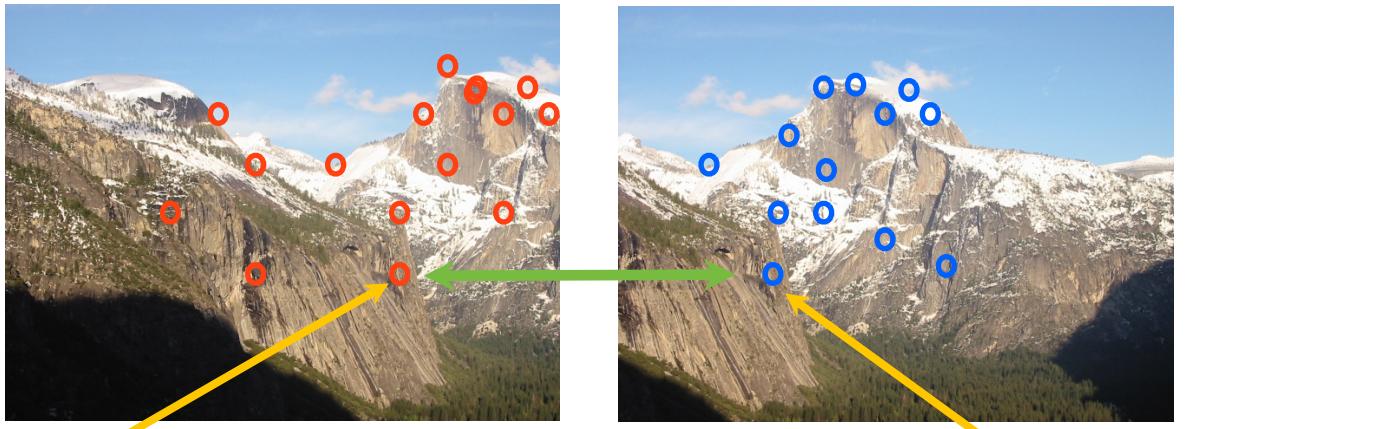
- Fit the transformation matrix

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ 0 & 0 & 1 \end{bmatrix}$$

- Six variables
  - each point give two equations
  - at least three points

- Least squares

Main Flow



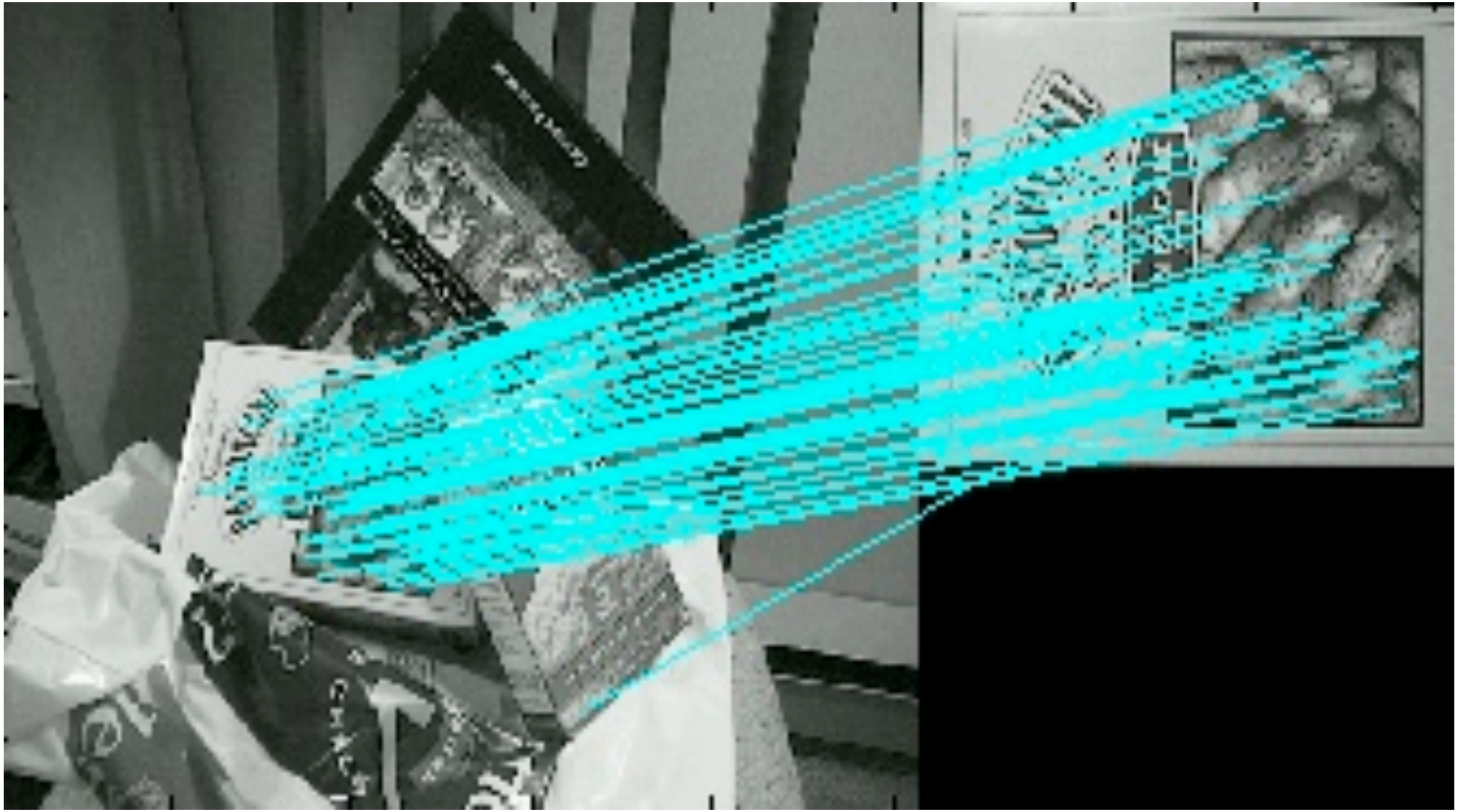
$(u_1, u_2, \ldots, u_{128})$

$(v_1, v_2, \ldots, v_{128})$

- Detect key points
- Build the SIFT descriptors
- Match SIFT descriptors
- Fitting the transformation
- RANSAC

RANSAC

- A further refinement of matches

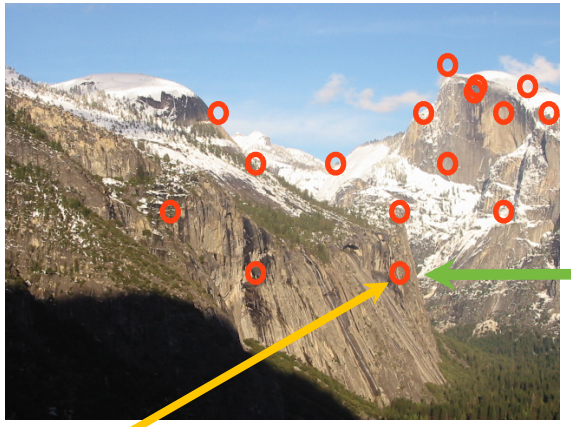


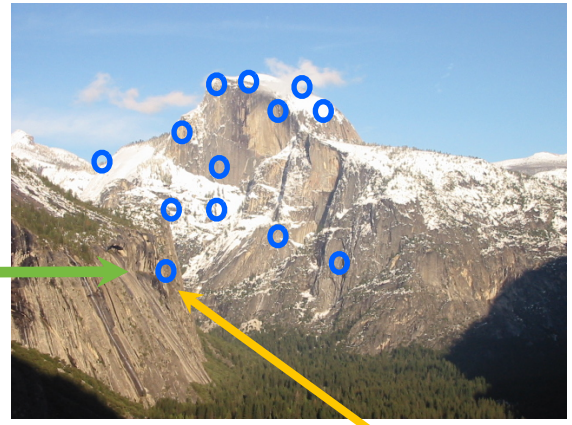
Skeleton Code

- RANSAC
  - ComputeError

$$\left\| \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} - H \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \right\|_2$$

Main Flow



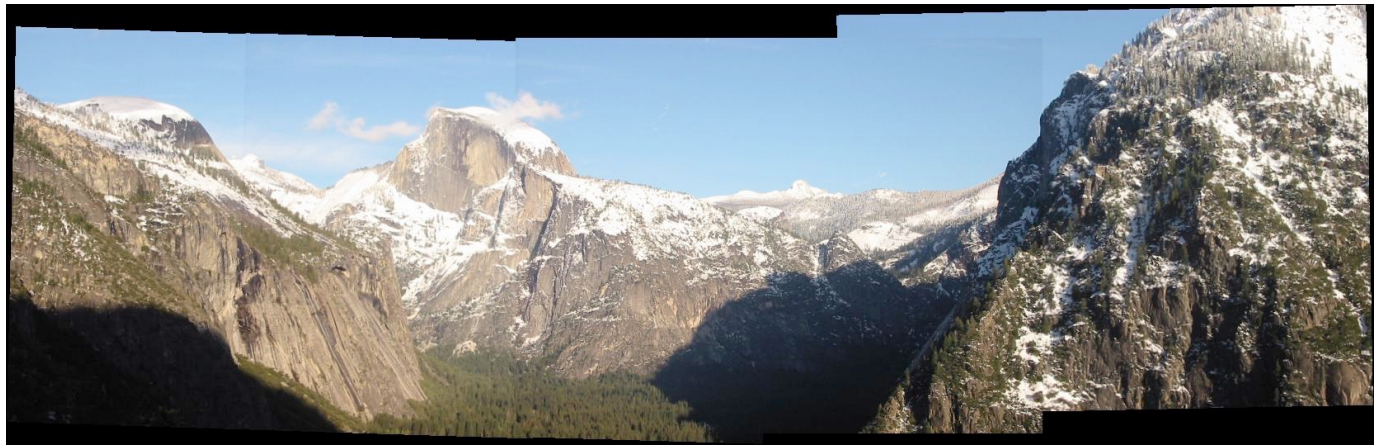


$(u_1, u_2, \ldots, u_{128})$

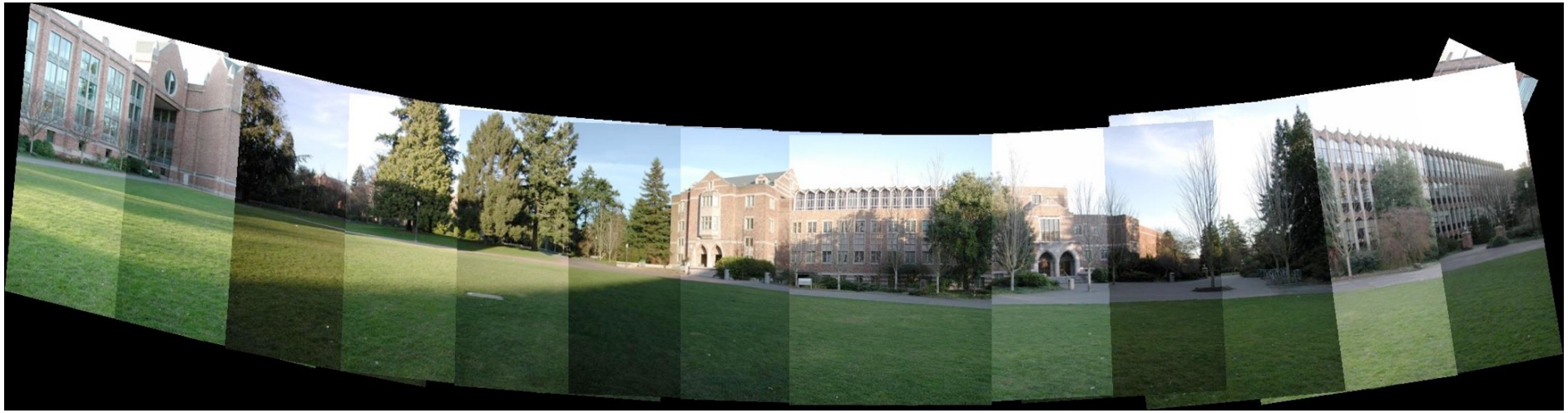
$(v_1, v_2, \ldots, v_{128})$

- Detect key points
- Build the SIFT descriptors
- Match SIFT descriptors
- Fitting the transformation
- RANSAC

Results



Results



What we have learned this week?

- SIFT: an image region descriptor
- HOG: another image descriptor
- Other image descriptors

- Matching [Read Szeliski 4.1]
- Application: Panorama

Some background reading: R. Szeliski, Ch 4.1; David Lowe, IJCV 2004

Thanks