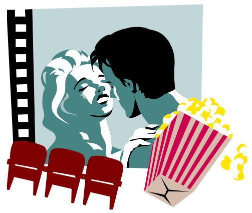
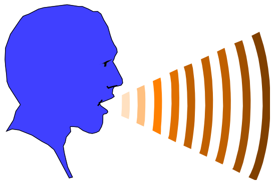



Image and Video Processing

Feature Detection and Feature Descriptors

Yao Wang
Tandon School of Engineering, New York University

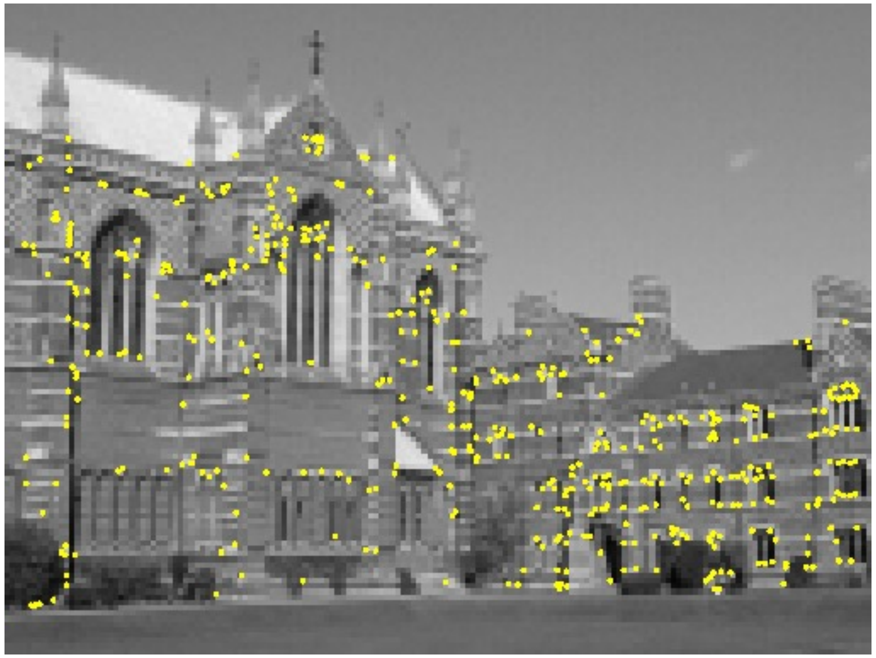
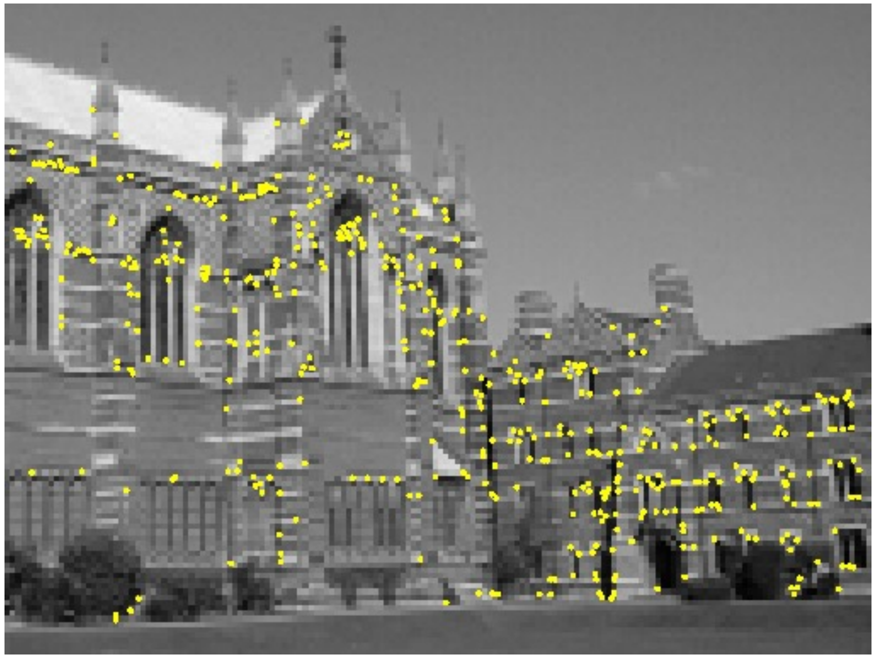
Lecture Outline

- ⇨ Need for Features and Feature Descriptors
- Feature Detectors
  - Desired properties of features
  - Harris Detector
  - Scale Space
  - Harris-Laplace detector
  - SIFT detector
  - Scale and orientation detection
- Feature descriptors
  - SIFT
  - SURF
- Deep learning for feature detection and description
- Image classification using Bag of Visual Words

Why do we want to detect ad describe features?

- To establish correspondence between two images
  - Stitching of panorama
  - Image registration based on feature correspondence
- To describe an entire image or object for classification
  - Based on properties of all features detected in an image or object
  - Through the idea of bag of visual words

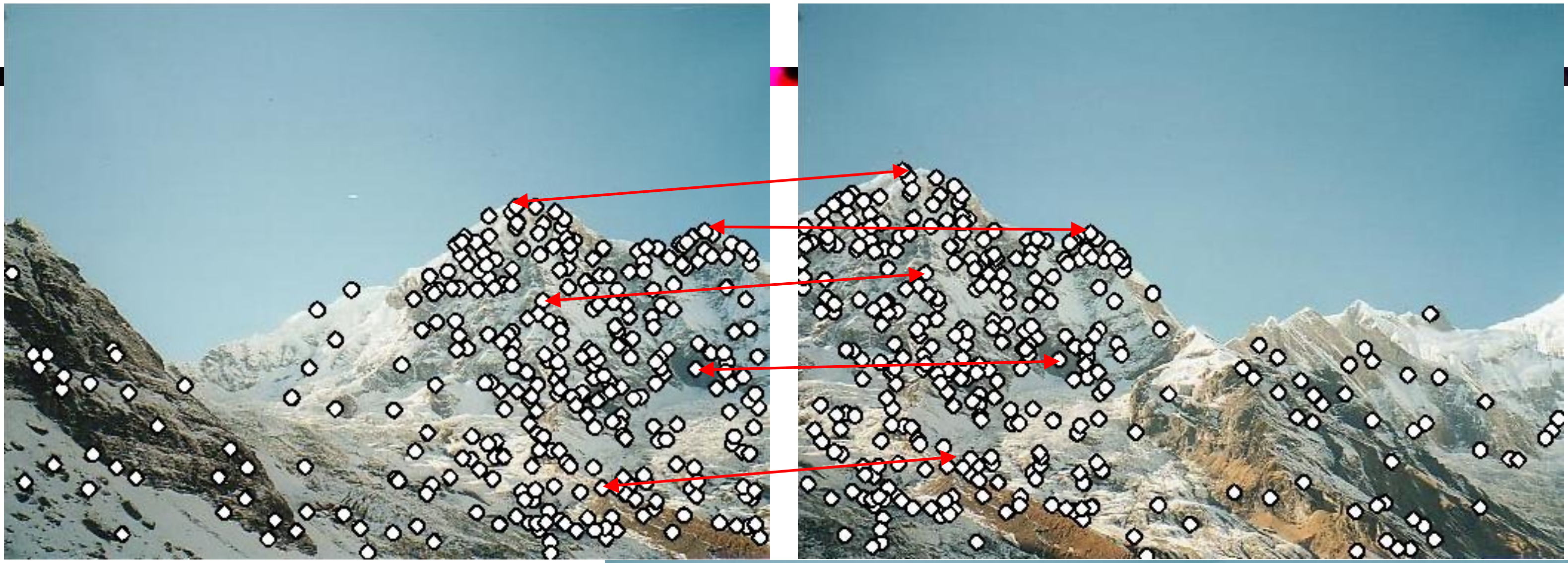
Feature Correspondences

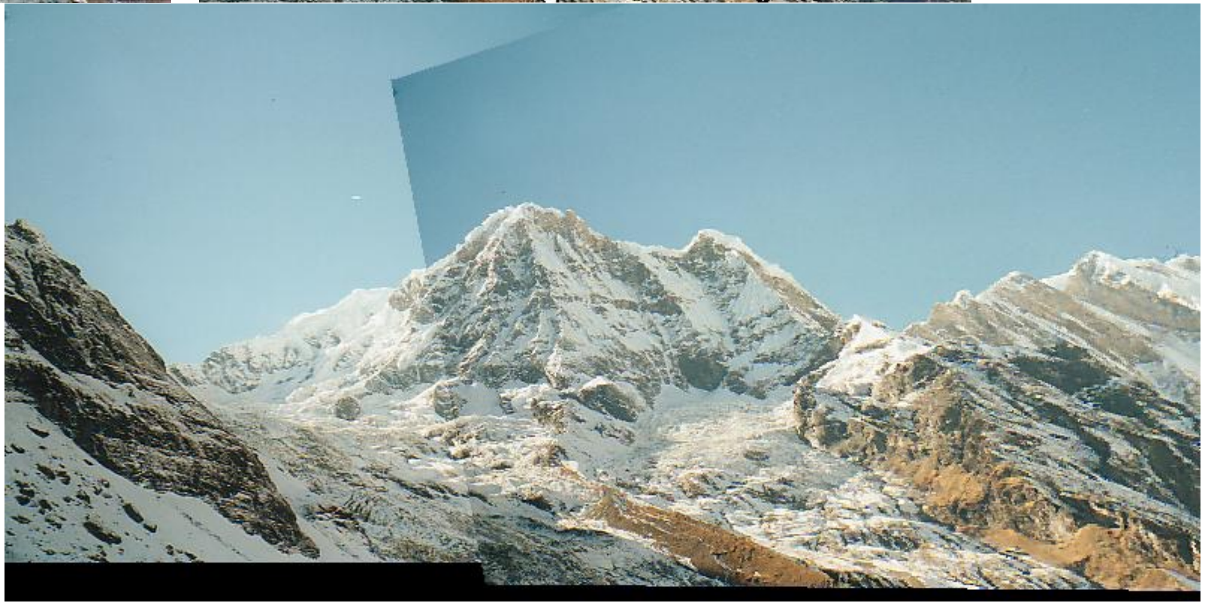



Interest points extracted with Harris Detector (~ 500 points)

From https://courses.cs.washington.edu/courses/cse455/16wi/notes/index.html, lecture on interest operators

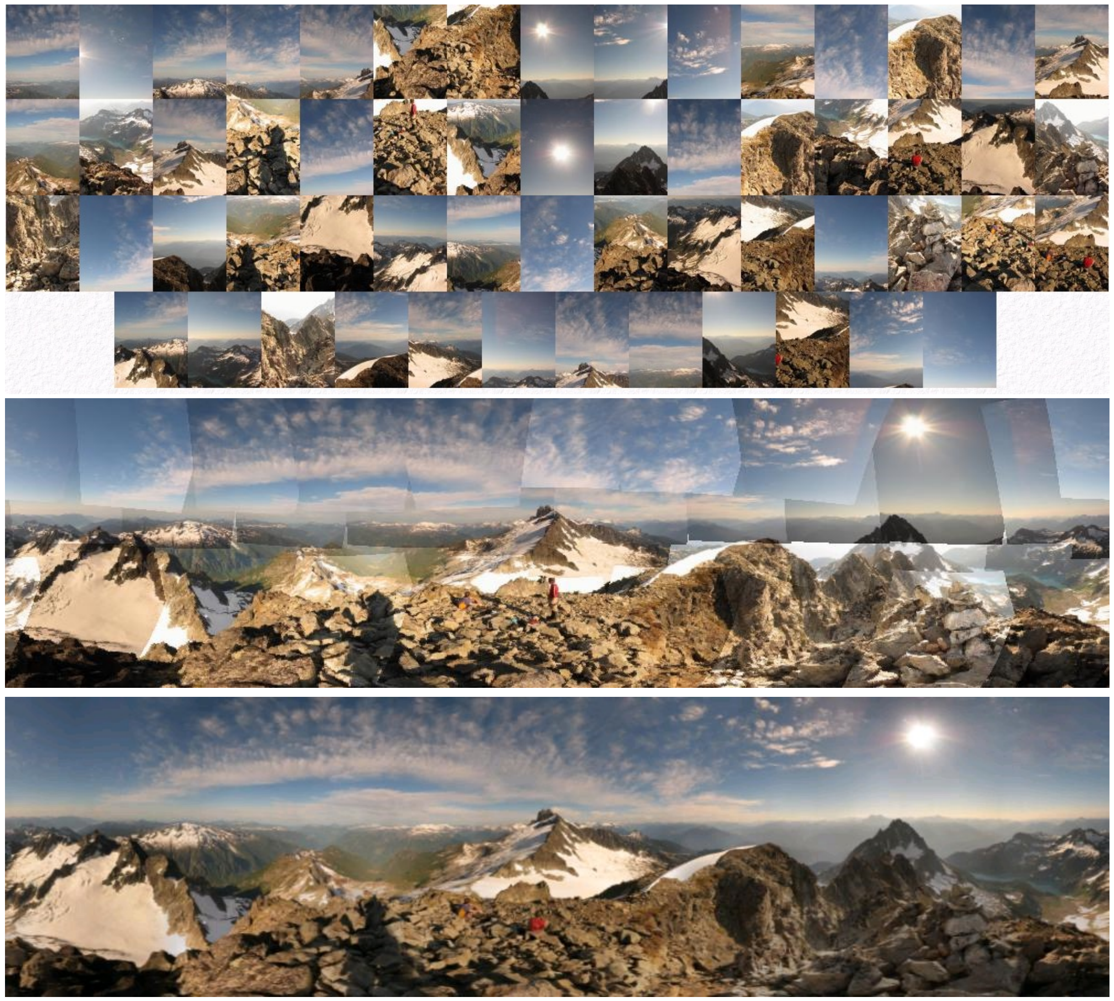
Panorama Stitching





From https://courses.cs.washington.edu/courses/cse455/16wi/notes/index.html, lecture on descriptors

Panorama Stitching



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

From https://courses.cs.washington.edu/courses/cse455/16wi/notes/index.html, lecture on descriptors

Lecture Outline

- Need for Features and Feature Descriptors
- Feature Detectors
  - ⇨ Desired properties of features
  - Harris Detector
  - Scale Space
  - Harris-Laplace detector
  - SIFT detector
  - Scale and orientation detection
- Feature descriptors
  - SIFT
  - SURF
- Deep learning for feature detection and description
- Image classification using Bag of Visual Words

What Features Are Good?

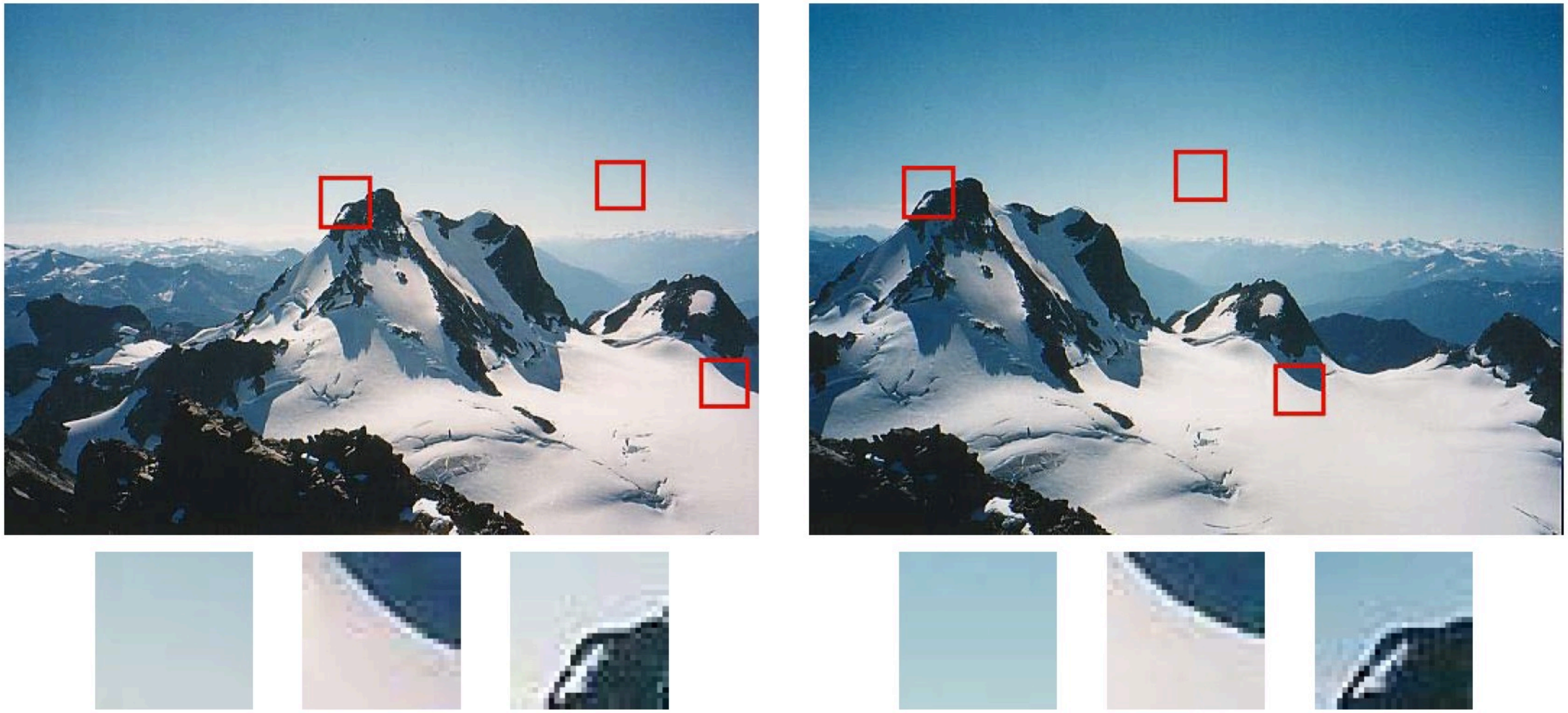
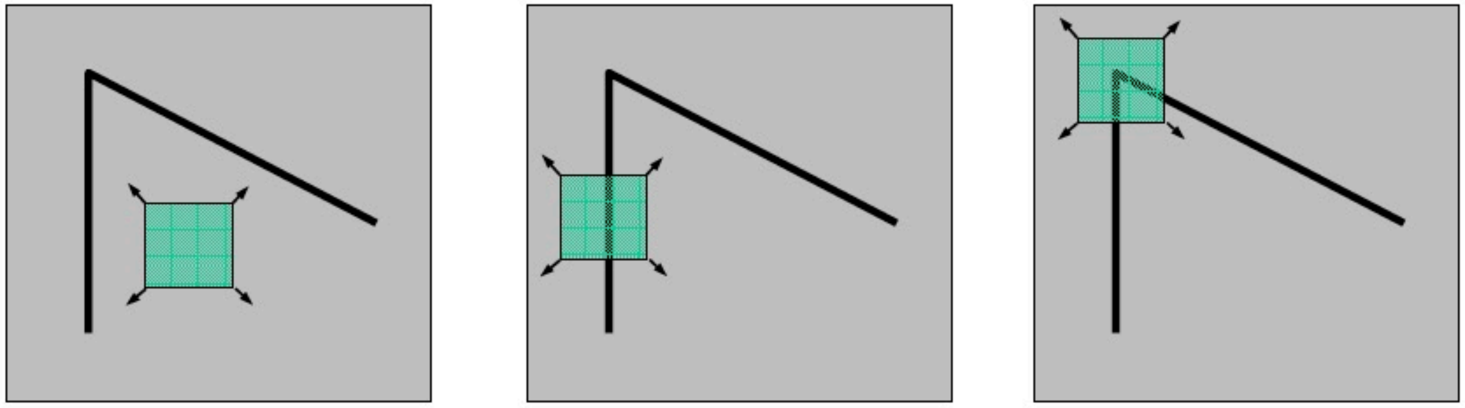


Figure 4.3 Image pairs with extracted patches below. Notice how some patches can be localized or matched with higher accuracy than others.

From Szeliski, *Computer Vision: Algorithms and Applications*, 2010

What features are good?



"flat" region:
no change in all directions

"edge":
no change along the edge direction

"corner":
significant change in all directions

Want to detect corners!

From Szeliski, *Computer Vision: Algorithms and Applications*, 2010

Mathematical Formulation

- We would like a feature to be located where a slight shift in any direction causes a large difference, i.e. we want the following energy function to be large for any $\Delta u=[\Delta x, \Delta y]^T$:

$$E_{\mathrm{AC}}(\Delta \boldsymbol{u}) = \sum_i w(\boldsymbol{x}_i)[I_0(\boldsymbol{x}_i + \Delta \boldsymbol{u}) - I_0(\boldsymbol{x}_i)]^2$$

- Sum is over a window centered at the point being examined.
- W(x) is usually a Gaussian function

Taylor Series for 2D Functions

$$f(x+u,y+v) = f(x,y) + uf_x(x,y) + vf_y(x,y) +$$

First partial derivatives

$$\frac{1}{2!}\left[u^2 f_{xx}(x,y) + uv f_{xy} x,y + v^2 f_{yy}(x,y)\right] +$$

Second partial derivatives

$$\frac{1}{3!}\left[u^3 f_{xxx}(x,y) + u^2 v f_{xxy}(x,y) + uv^2 f_{xyy}(x,y) + v^3 f_{yyy}(x,y)\right]$$

Third partial derivatives

$+ \ \ldots$ (Higher order terms)

First order approx

$$f(x+u,y+v) \approx f(x,y) + uf_x(x,y) + vf_y(x,y)$$

http://www.cse.psu.edu/~rtc12/CSE486/lecture06.pdf

Taylor Expansion and Moment Matrix

$$\begin{aligned}
E_{\mathrm{AC}}(\Delta \boldsymbol{u}) &= \sum_i w(\boldsymbol{x}_i)[I_0(\boldsymbol{x}_i + \Delta \boldsymbol{u}) - I_0(\boldsymbol{x}_i)]^2 \\
&\approx \sum_i w(\boldsymbol{x}_i)[I_0(\boldsymbol{x}_i) + \underbrace{\nabla I_0(\boldsymbol{x}_i) \cdot \Delta \boldsymbol{u}}_{I_x \Delta x + I_y \Delta y} - I_0(\boldsymbol{x}_i)]^2 \\
&= \sum_i w(\boldsymbol{x}_i)[\nabla I_0(\boldsymbol{x}_i) \cdot \Delta \boldsymbol{u}]^2 \\
&= \Delta \boldsymbol{u}^T \boldsymbol{A} \Delta \boldsymbol{u},
\end{aligned}$$

$$A = \begin{bmatrix} \sum_x w(x) I_x^2 & \sum_x w(x) I_x I_y \\ \sum_x w(x) I_x I_y & \sum_x w(x) I_y^2 \end{bmatrix} \qquad \nabla I = \begin{bmatrix} I_x \\ I_y \end{bmatrix}, \quad \Delta u = \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

A is called Moment Matrix

Using eigenvalues of A to evaluate the goodness of a feature point

Eigenvectors ϕ_i and eigenvalues λ_i of A are defined as

$\mathrm{A}\phi_i = \lambda_i \phi_i, \quad i = 0,1$ ϕ_0 and ϕ_1 are orthonormal

Once ϕ_i and λ_i are found, A can be represented as

$$A = \begin{bmatrix} \phi_1 & \phi_0 \end{bmatrix} \begin{bmatrix} \lambda_1 & \\ & \lambda_0 \end{bmatrix} \begin{bmatrix} \phi_1^T \\ \phi_0^T \end{bmatrix}$$

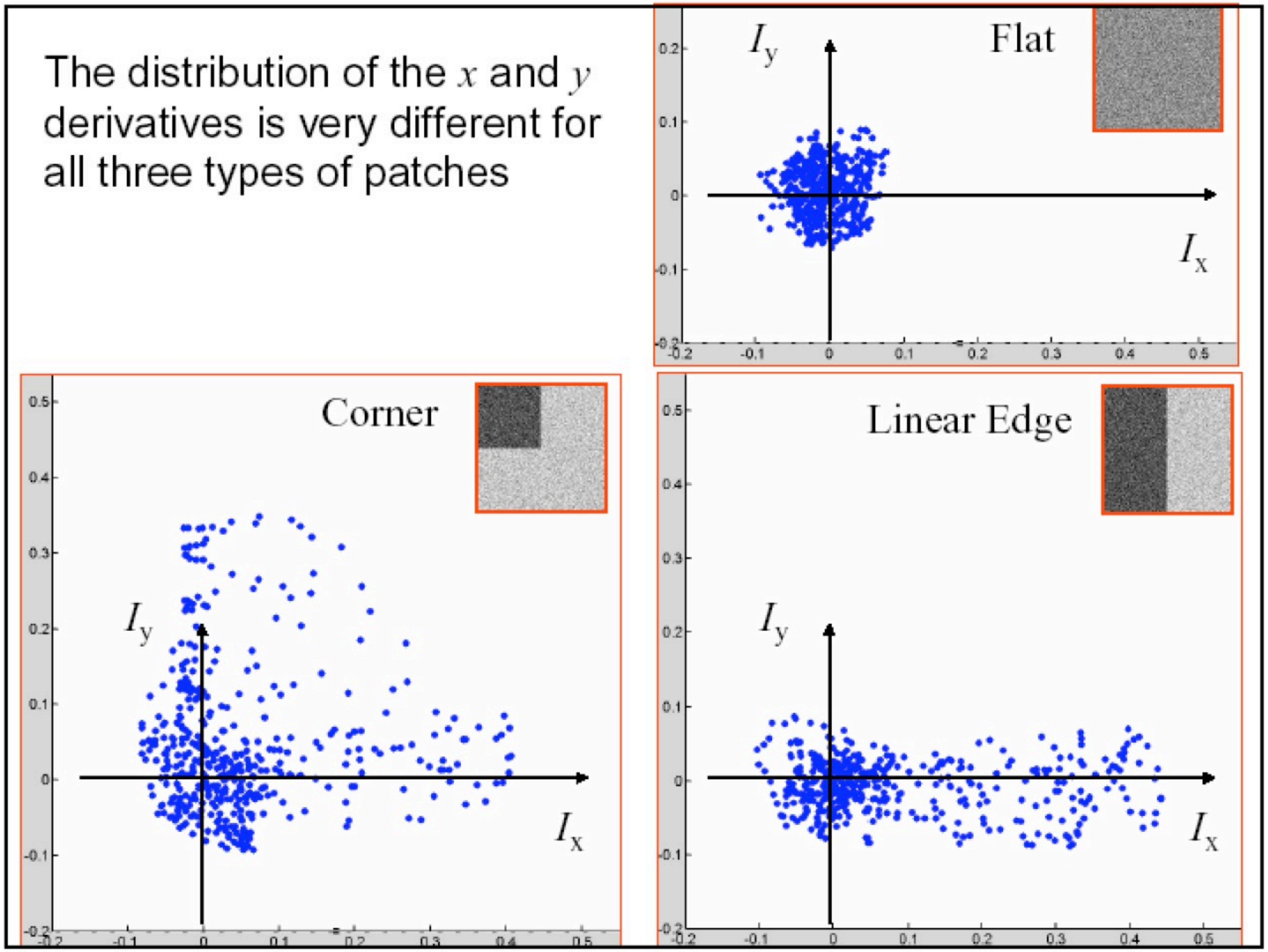
Assuming $\lambda_0 <= \lambda_1$ and if $\Delta \mathrm{u} = \phi_0$,

$$E(\phi_0) = \phi_0^T \begin{bmatrix} \phi_1 & \phi_0 \end{bmatrix} \begin{bmatrix} \lambda_1 & \\ & \lambda_0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \phi_0^T \begin{bmatrix} \phi_1 & \phi_0 \end{bmatrix} \begin{bmatrix} 0 \\ \lambda_0 \end{bmatrix} = \lambda_0$$

- The eigenvector ϕ_0 is the direction with the minimal change
- We want to find features points where the minimal eigenvalue is large! = we want both eigenvalues to be large!

Robert Collins
CSE486, Penn State

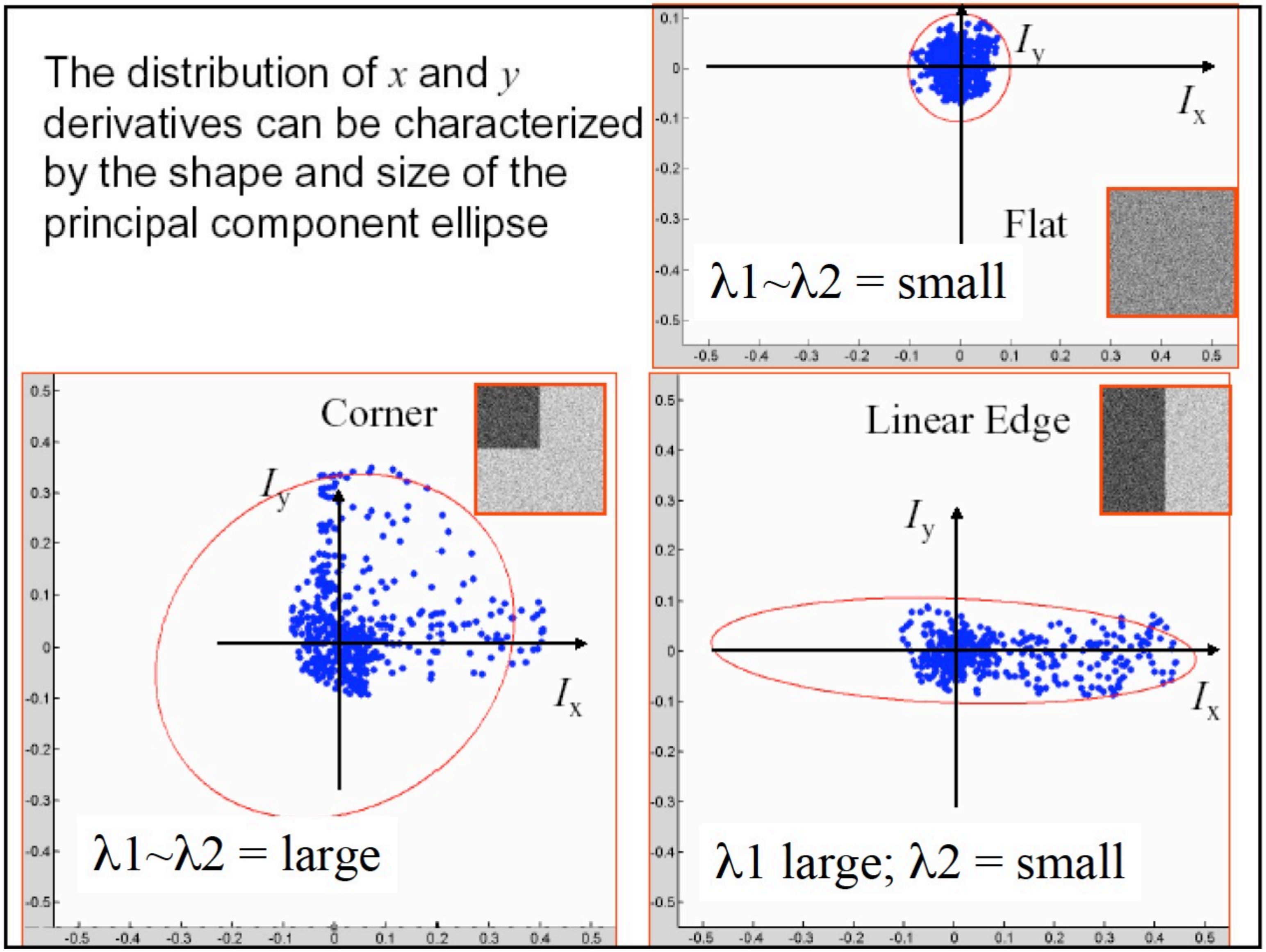
Plotting Derivatives as 2D Points


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

From: http://www.cse.psu.edu/~rtc12/CSE486/lecture06.pdf

Fitting Ellipse to each Set of Points

The distribution of x and y derivatives can be characterized by the shape and size of the principal component ellipse



http://www.cse.psu.edu/~rtc12/CSE486/lecture06.pdf

Harris Detector (aka Corner Detector)

- Feature points should be where both eigenvalues are large!
- How do we quantify when both eigenvalues are large?
  - Given two numbers, λ_0, λ_1 , geometric mean $= \sqrt{\lambda_0\lambda_1}$, arithmetic mean $= (\lambda_0 + \lambda_1)$/2 .
  - Generally arithmetic mean>geometric mean. Equal only if $\lambda_0 = \lambda_1$
    - Ex: $\lambda_0 = 1, \lambda_1 = 9$
  - Difference between geometric mean $\sqrt{\lambda_0\lambda_1}$ and arithmetic mean $(\lambda_0 + \lambda_1)$/2 should be large!

- Harris detector interest measure:

$$H = \lambda_0\lambda_1 - \alpha(\lambda_0 + \lambda_1)^2, \quad \alpha = 0.06$$

C.Harris and M.Stephens. “A Combined Corner and Edge Detector.” Proceedings of the 4th Alvey Vision Conference: pages 147—151, 1988.

How to compute Harris value?

$$\text{Because } A = \begin{bmatrix} \phi_1 & \phi_0 \end{bmatrix} \begin{bmatrix} \lambda_1 & \\ & \lambda_0 \end{bmatrix} \begin{bmatrix} \phi_1^T \\ \phi_0^T \end{bmatrix} = \begin{bmatrix} a_{00} & a_{01} \\ a_{01} & a_{11} \end{bmatrix}$$

$$\text{Det}(A) = \lambda_1 \lambda_0 = a_{00} a_{11} - a_{01}^2$$

$$\text{Trace}(A) = \lambda_1 + \lambda_0 = a_{00} + a_{11}$$

$$\text{Therefore } \ \text{H=Det}(A) - \alpha \text{Trace}(A)^2$$

No need to calculate the eigenvalues!

Forming the Moment Matrix

- Recall that to form the A-matrix for each pixel, we need to form a neighborhood window and apply Gaussian weighting (usually with σ=2) to the I_x^2, I_y^2, and I_xI_y values in the neighborhood and sum.

$$A = \begin{bmatrix} \sum_x w(x) I_x^2 & \sum_x w(x) I_x I_y \\ \sum_x w(x) I_x I_y & \sum_x w(x) I_y^2 \end{bmatrix}$$

- This can be instead done by convolving the images of I_x^2, I_y^2, and I_xI_y by the Gaussian filter first, to generate images denoted by A_{xx}, A_{yy}, A_{xy}. Then the values in each convolved images at each pixel can be directly used in computing the A matrix for that pixel.

$$A = \begin{bmatrix} A_{xx} & A_{xy} \\ A_{xy} & A_{yy} \end{bmatrix}$$

Harris Corner Point Detection Algorithm

- Compute the horizontal and vertical gradient images I_x and I_y by convolving the original image with derivative of Gaussian filters H_x and H_y (usually with $\sigma=1$)
- Form three images I_x^2, I_y^2, and I_xI_y
- Convolve the images of I_x^2, I_y^2, and I_xI_y by a Gaussian filter (usually with $\sigma=2$) to generate images of A_{xx}, A_{yy}, A_{xy}.
- For each pixel, form the A matrix, and determine the Harris interest value $$A = \begin{bmatrix} A_{xx} & A_{xy} \\ A_{xy} & A_{yy} \end{bmatrix} \qquad H = \det(A) - \alpha \ (Trace(A))^2$$
- Repeat the above process for each pixel to form the Harris interest measure image (Harris image)
- Find all local maxima in the Harris image and remove those points where the interest value < a threshold. Alternatively, you can keep the N pixels with largest N Harris values.
- The process of finding local maxima is also known as non-maximal suppression.

Filter for Taking Derivatives (Review)

- Apply Gaussian filtering first to smooth the image, STD depends on noise level or desired smoothing effect
- Then take derivative in horizontal and vertical directions
- = Convolve the image with a Derivative of Gaussian filter

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

$$H_x(x,y)=\frac{\partial G}{\partial x}=-\frac{x}{\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$H_y(x,y)=\frac{\partial G}{\partial y}=-\frac{y}{\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}$$

- Sample the above continuous filter to get digital filter. Hy is rotated version of Hx

Gaussian Filter (Review)

- Analog form: STD σ controls the smoothing strength

$$G(x,y) = \exp\left\{-\frac{x^2+y^2}{2\sigma^2}\right\},$$

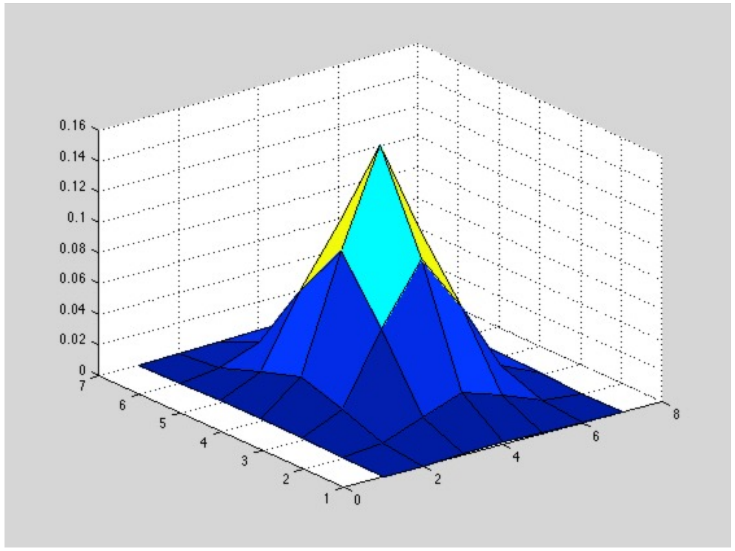
- Take samples, truncate after a few STD, normalize the sum to 1. Usually σ>=1
- Size of mask nxn, typically n>=2σ+1 (2σ on each side), Ex: σ=1, n=7.
  - Show filter mask,
  - Show frequency response
- Essentially a weighted average filter with decreasing weights away from the center
- A good filter for removing noise. σ should be chosen based on noise STD.
- Can also be applied to an image patch pixel-wise as a weighting map.

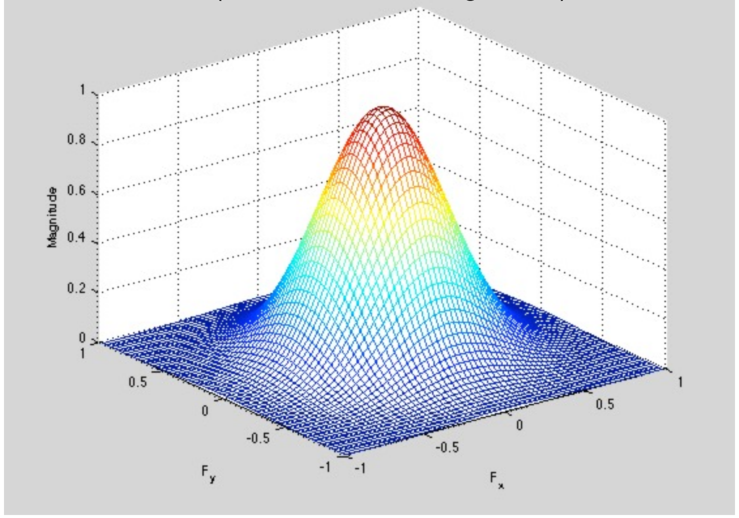
Ex: 7x7 Gaussian Filter Generation

```
0.0000    0.0002    0.0011    0.0018    0.0011    0.0002    0.0000
0.0002    0.0029    0.0131    0.0216    0.0131    0.0029    0.0002
0.0011    0.0131    0.0586    0.0966    0.0586    0.0131    0.0011
0.0018    0.0216    0.0966    0.1592    0.0966    0.0216    0.0018
0.0011    0.0131    0.0586    0.0966    0.0586    0.0131    0.0011
0.0002    0.0029    0.0131    0.0216    0.0131    0.0029    0.0002
0.0000    0.0002    0.0011    0.0018    0.0011    0.0002    0.0000
```

```
function gauss(s)

x=[-3.0:1.0:3.0];
gauss=exp(-x.^2/(2*s^2));
gauss2=gauss'*gauss;
gauss2=gauss2/(sum(sum(gauss2)));
H=gauss2;
disp(H);
figure(1);
surf(gauss2);
figure(2);
freqz2(gauss2);
title('Gaussian Filter \sigma=1');
```





Derivative Filter Examples

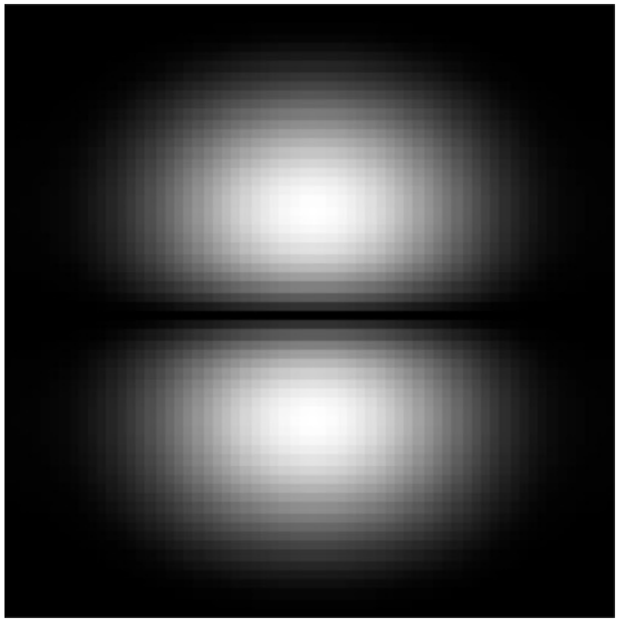
σ=1, n=5

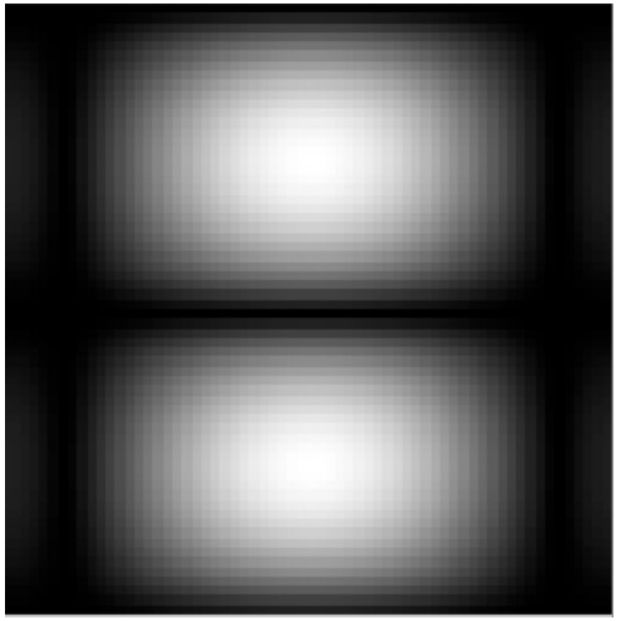
0.0366	0.1642	0.2707	0.1642	0.0366
0.0821	0.3679	0.6065	0.3679	0.0821
0	0	0	0	0
-0.0821	-0.3679	-0.6065	-0.3679	-0.0821
-0.0366	-0.1642	-0.2707	-0.1642	-0.0366

σ =1, n=3 (similar to Sobel)

0.3679	0.6065	0.3679
0	0	0
-0.3679	-0.6065	-0.3679

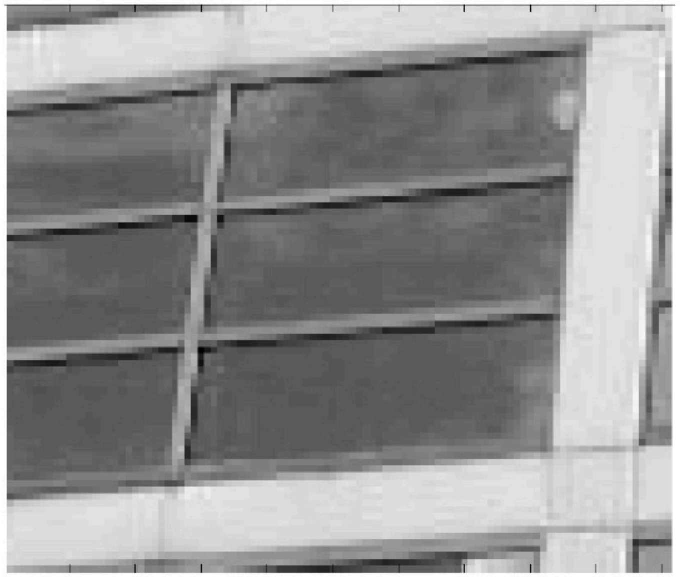
Frequency response of filter

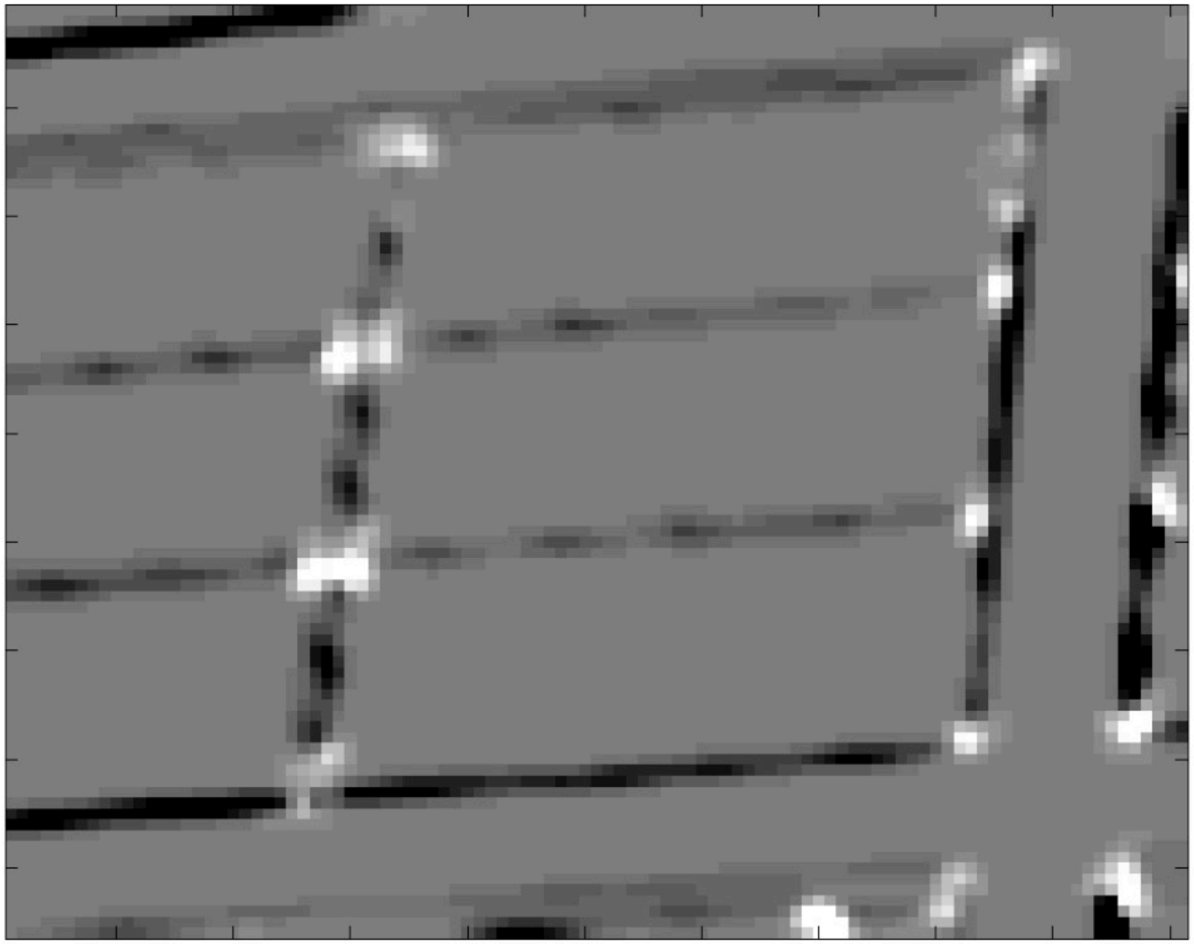




Low-pass along the edge, and band-pass in orthogonal direction (across edge)

Corner Response Example





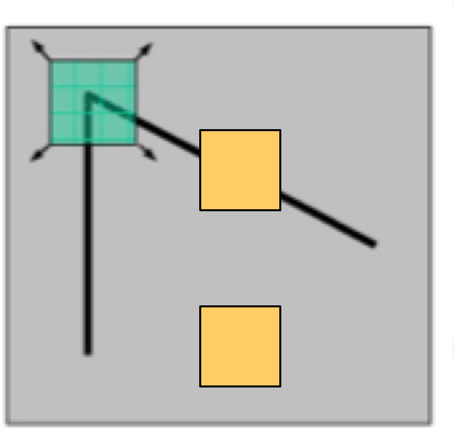
Harris R score.

Ix, Iy computed using Sobel operator
Windowing function w = Gaussian, sigma=1

http://www.cse.psu.edu/~rtc12/CSE486/lecture06.pdf

Pop Quiz!

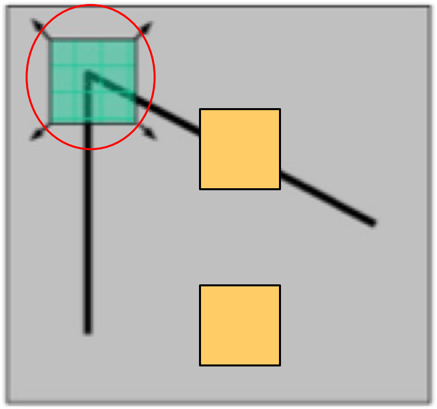
- Which pixels (center of different patches) are good features?



- What does Harris detector detects?
- How to determine the moment matrix A?
- What do the eigenvectors and eigenvalues of A indicate?

Pop Quiz!

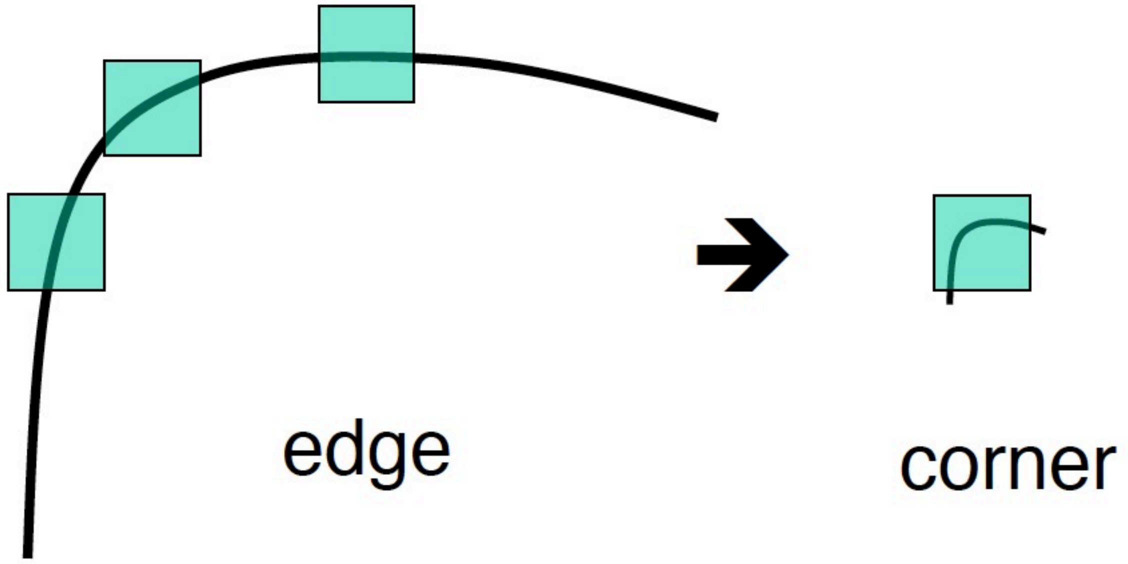
- Which pixels (center of different patches) are good features?



- What does Harris detector detects?
  - Corners
- How to determine the moment matrix A?
  - Determine the gradient images I_x, I_y by convolving with appropriate filters
  - Generate images of $I_x^2, I_x^2, I_x I_y$
  - Form the moment matrix at every pixel based on the $I_x^2, I_x^2, I_x I_y$ values of its neighborhood
- What do the eigenvectors and eigenvalues of A indicate?
  - Eigen vector corresponding to the largest eigenvalue denotes the direction with the largest variation

How good is Harris Detector?

- Invariant to brightness offset. Why?
- Invariant to shift and rotation. Why?
- Not invariant to spatial scaling!



Other detectors

- Vary in terms of the interest measure
- Hessian Detector:
  - Form a Hessian matrix (Using 2nd order gradients)
  - Using the Determinant of the Hessian matrix as the interest measure

$$\mathbf{H}[x,y] = \begin{bmatrix} f_{xx}[x,y] & f_{xy}[x,y] \\ f_{xy}[x,y] & f_{yy}[x,y] \end{bmatrix} = \begin{bmatrix} D_{xx}[x,y] * f[x,y] & D_{xy}[x,y] * f[x,y] \\ D_{xy}[x,y] * f[x,y] & D_{yy}[x,y] * f[x,y] \end{bmatrix}$$

$$\det \mathbf{H}[x,y] = f_{xx}[x,y] f_{yy}[x,y] - \left(f_{xy}[x,y]\right)^2$$

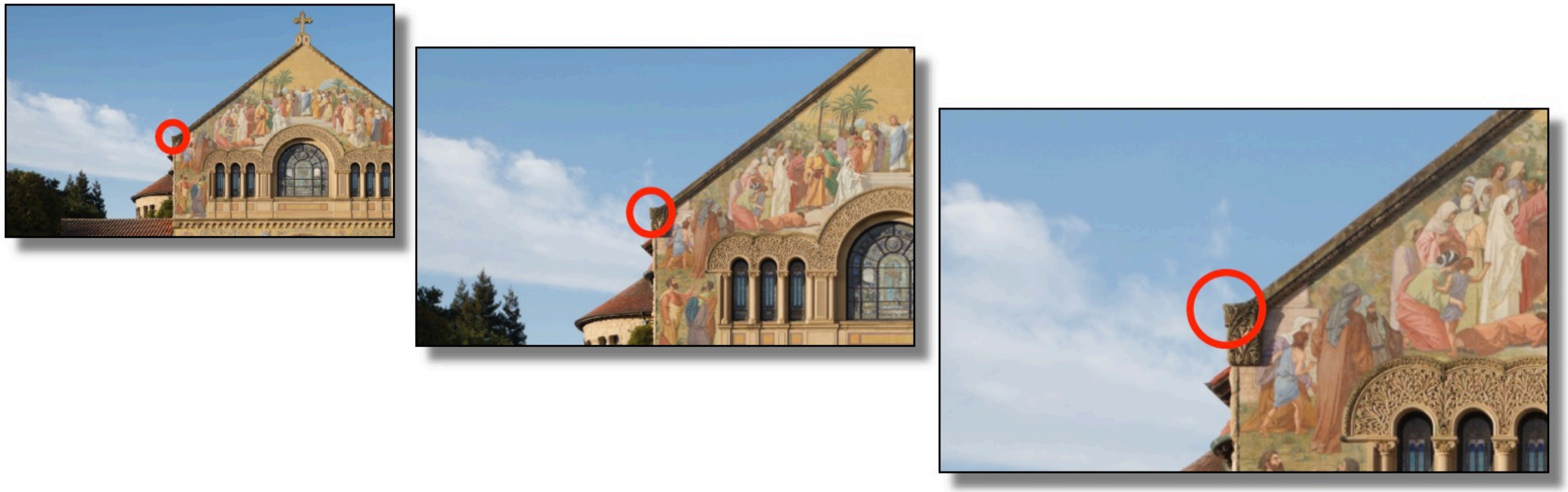
- Maximally Stable Extremal Regions
- Problem: second order gradient is sensitive to noise

Lecture Outline

- Need for Features and Feature Descriptors
- Feature Detectors
  - Desired properties of features
  - Harris Detector
  - ⇨ Scale Space
  - Harris-Laplace detector
  - SIFT detector
  - Scale and orientation detection
- Feature descriptors
  - SIFT
  - SURF
- Deep learning for feature detection and description
- Image classification using Bag of Visual Words

Scale-space image processing

- Corresponding image features can appear at different scales

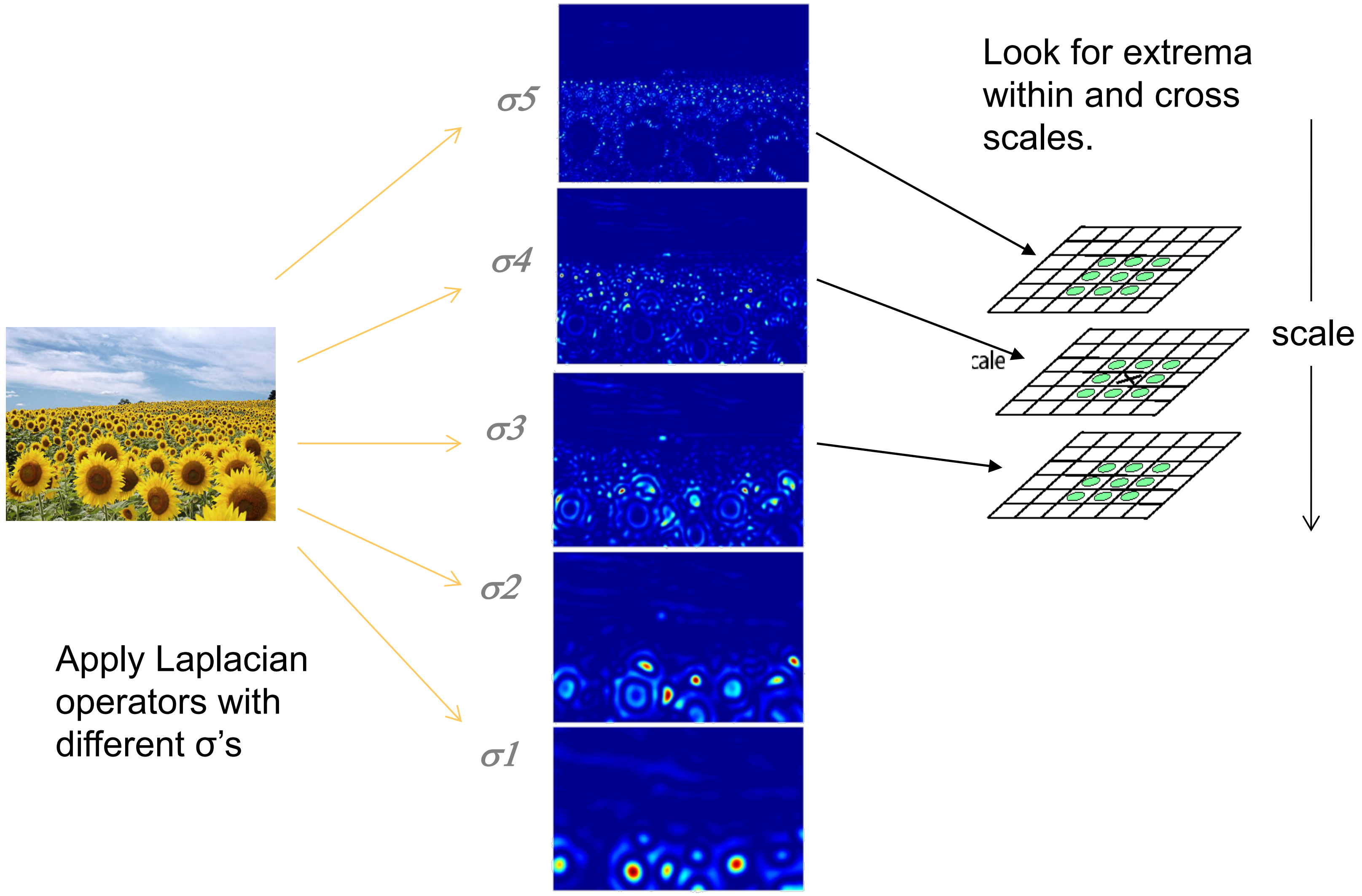


- Like shift-invariance, ***scale-invariance*** of image processing algorithms is often desirable.
- Scale-space representation is useful to process an image in a manner that is both shift-invariant and scale-invariant

[courtesy B. Girod, (c) 2013 Stanford University]

From http://web.stanford.edu/class/ee368/Handouts/Lectures/2016_Autumn/14-ScaleSpace_16x9.pdf

Local maxima of magnitude of normalized Laplacian images in both position and scale are good feature points!



Look for extrema within and cross scales.

Apply Laplacian operators with different σ's

From https://courses.cs.washington.edu/courses/cse455/16wi/notes/index.html, lecture on interest operators

Derivative and Laplacian of Gaussian

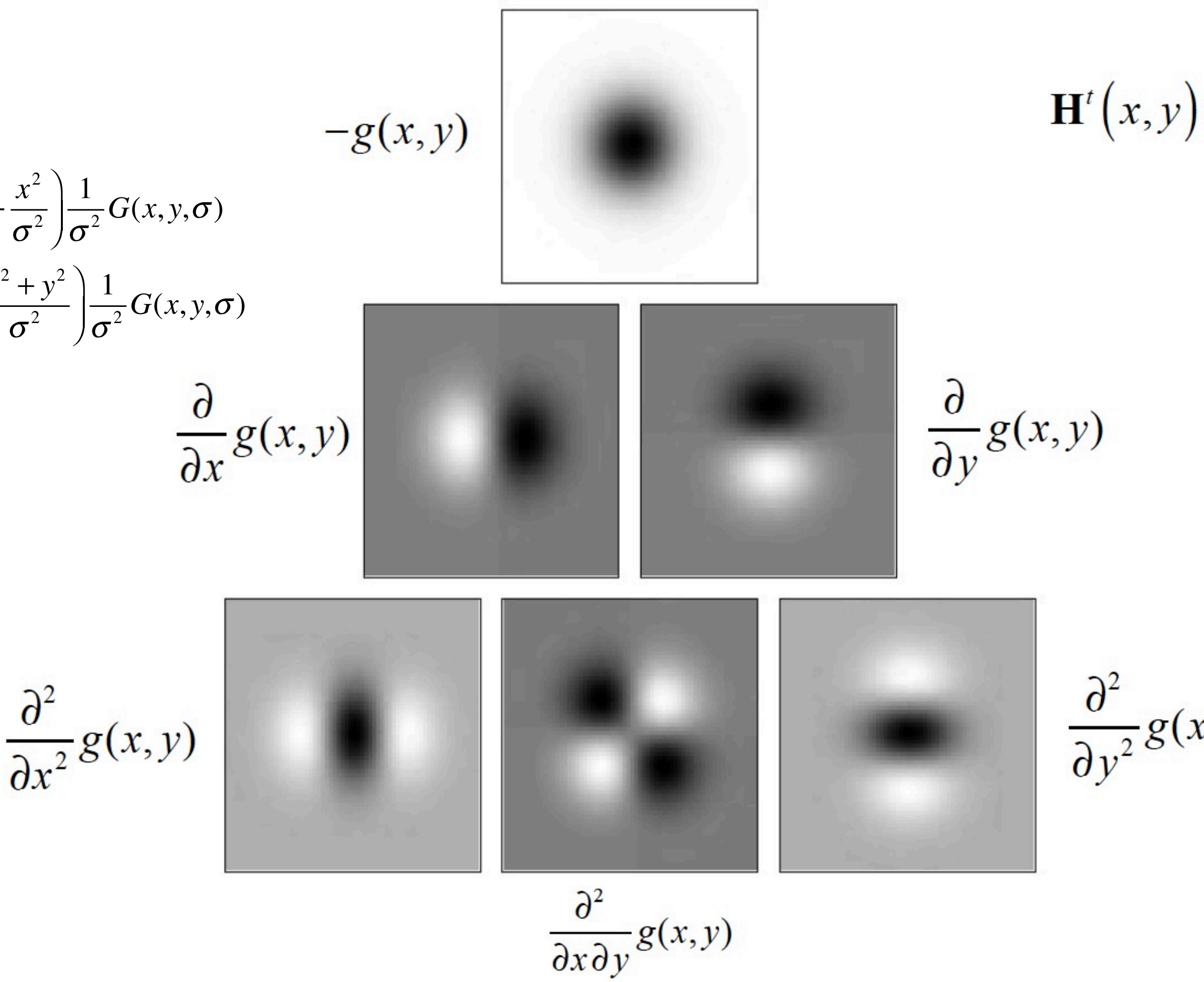
$$G(x,y,\sigma)=\frac{1}{2\pi\sigma^2}e^{-(x^2+y^2)/2\sigma^2}$$

$$G_x(x,y,\sigma)=-\frac{x}{2\pi\sigma^4}e^{-(x^2+y^2)/2\sigma^2}$$

$$G_{xx}(x,y,\sigma)=-\left(1-\frac{x^2}{\sigma^2}\right)\frac{1}{2\pi\sigma^4}e^{-(x^2+y^2)/2\sigma^2}=-\left(1-\frac{x^2}{\sigma^2}\right)\frac{1}{\sigma^2}G(x,y,\sigma)$$

$$\nabla G(x,y,\sigma)=G_{xx}(x,y,\sigma)+G_{yy}(x,y,\sigma)=-\left(2-\frac{x^2+y^2}{\sigma^2}\right)\frac{1}{\sigma^2}G(x,y,\sigma)$$

(Laplacian operator)



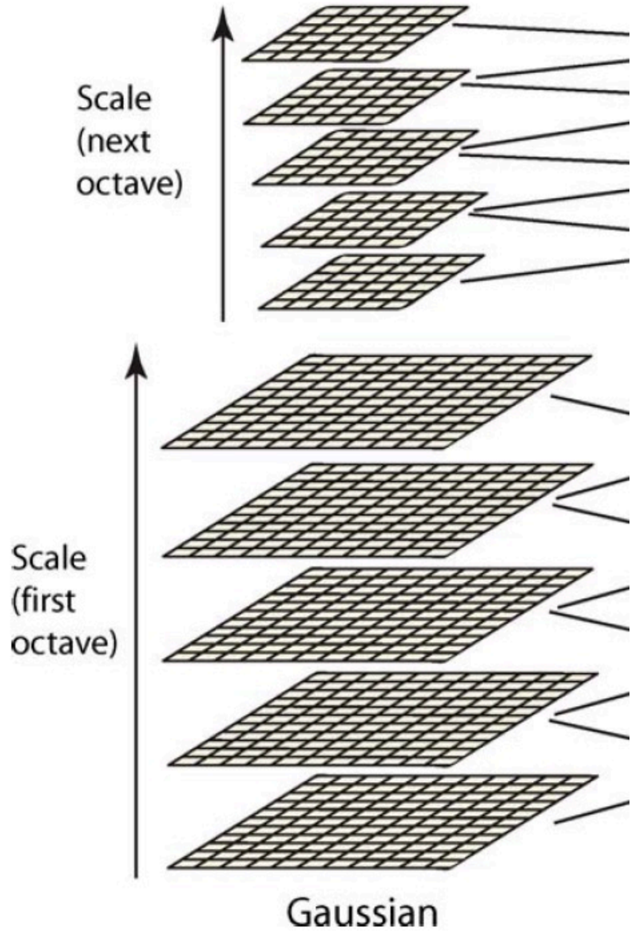
$-g(x,y)$
$\mathbf{H}^t(x,y)$
$\frac{\partial}{\partial x}g(x,y)$
$\frac{\partial}{\partial y}g(x,y)$
$\frac{\partial^2}{\partial x^2}g(x,y)$
$\frac{\partial^2}{\partial y^2}g(x$
$\frac{\partial^2}{\partial x\partial y}g(x,y)$

[courtesy B. Girod, (c) 2013 Stanford University]

Normalized Laplacian Operator for Feature Detection

- The local maxima of the absolute value of $\sigma^2\nabla^2 G$ (normalized Laplacian) produce the most stable image features compared to a range of other possible image functions, such as the gradient, Hessian, or Harris corner function.
- Can apply the Laplacian operator using different σ to detect feature points at different scales.

Gaussian Scale Space



A Gaussian pyramid with multiple scales at the same resolution:

$$L(x,y,\sigma_n)=G(x,y,\sigma_n)*I(x,y), \quad \sigma_n=\sigma_0 k^n$$

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

Typically, $\sigma_0=1, k=\sqrt{2}$

With the above choice, every octave has 3 scales

$n = 0,1,2$

In the left example, k=$2^{1/4}$, so each octave has 5 scales

$n = 0,1,2,3,4$

Using Difference of Gaussian to Approximate Normalized Laplacian

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

$$G_x(x,y,\sigma)=-\frac{x}{2\pi\sigma^4}e^{-(x^2+y^2)/2\sigma^2}$$

$$G_{xx}(x,y,\sigma)=-\left(1-\frac{x^2}{\sigma^2}\right)\frac{1}{2\pi\sigma^4}e^{-(x^2+y^2)/2\sigma^2}=-\left(1-\frac{x^2}{\sigma^2}\right)\frac{1}{\sigma^2}G(x,y,\sigma)$$

$$\nabla G(x,y,\sigma)=G_{xx}(x,y,\sigma)+G_{yy}(x,y,\sigma)=-\left(2-\frac{x^2+y^2}{\sigma^2}\right)\frac{1}{\sigma^2}G(x,y,\sigma)$$

It is easy to show that the Gaussian function satisfy the following equation

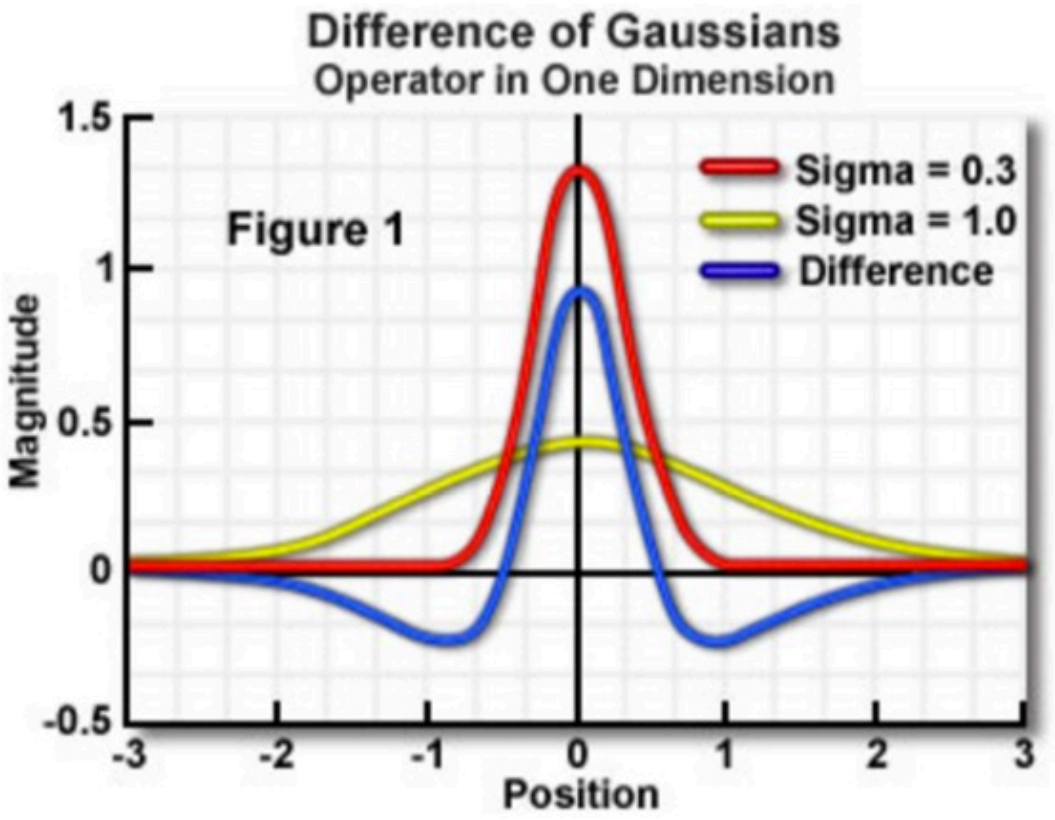
$$\frac{\partial G}{\partial\sigma}=\sigma\nabla^2G=\sigma\left(\frac{\partial^2G}{\partial x^2}+\frac{\partial^2G}{\partial y^2}\right)$$

Because $\frac{\partial G}{\partial\sigma}\approx\frac{G(x,y,k\sigma)-G(x,y,\sigma)}{k\sigma-\sigma}$

Therefore $G(x,y,\underset{k\sigma}{\cancel{t}\sigma})-G(x,y,\sigma)\approx(k-1)\sigma\frac{\partial G}{\partial\sigma}=(k-1)\sigma^2\nabla^2G$

This means that difference of Gaussian filtered images produces images of $\sigma^2\nabla^2G$ up to a constant scaling factor!

Using Difference of Gaussian (DoG) to Approximate Normalized Laplacian (LoG)



From https://courses.cs.washington.edu/courses/cse455/13au/: Lecture on “interest points”

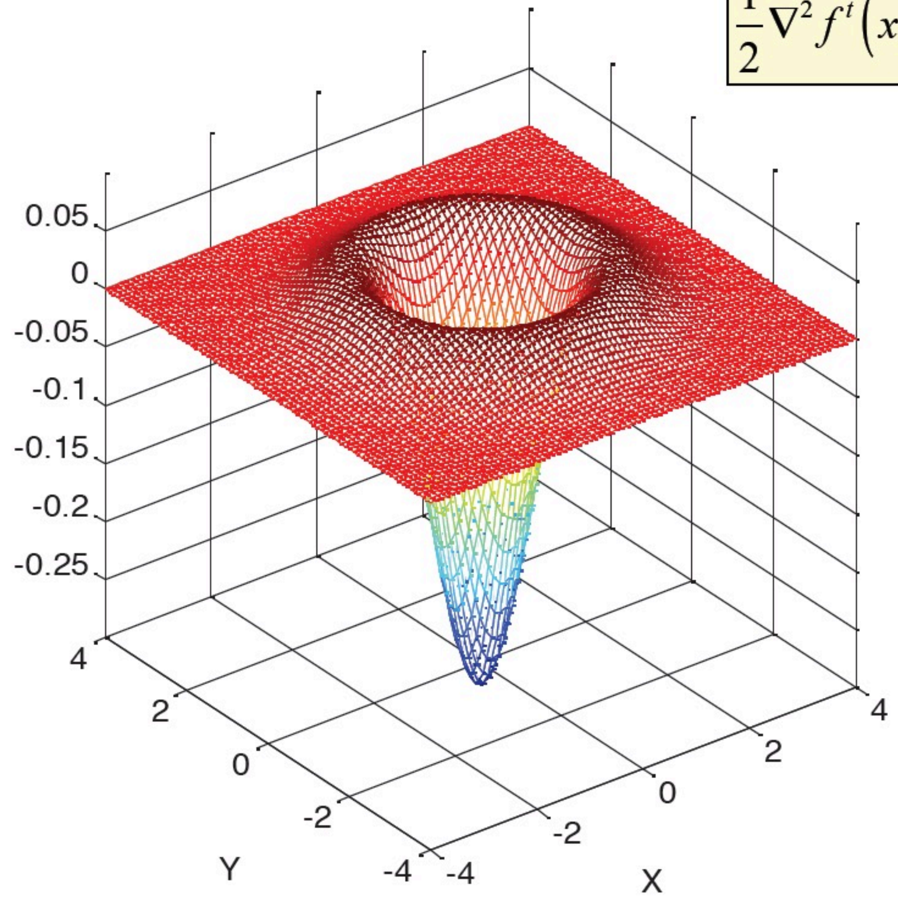
Laplacian of Gaussian

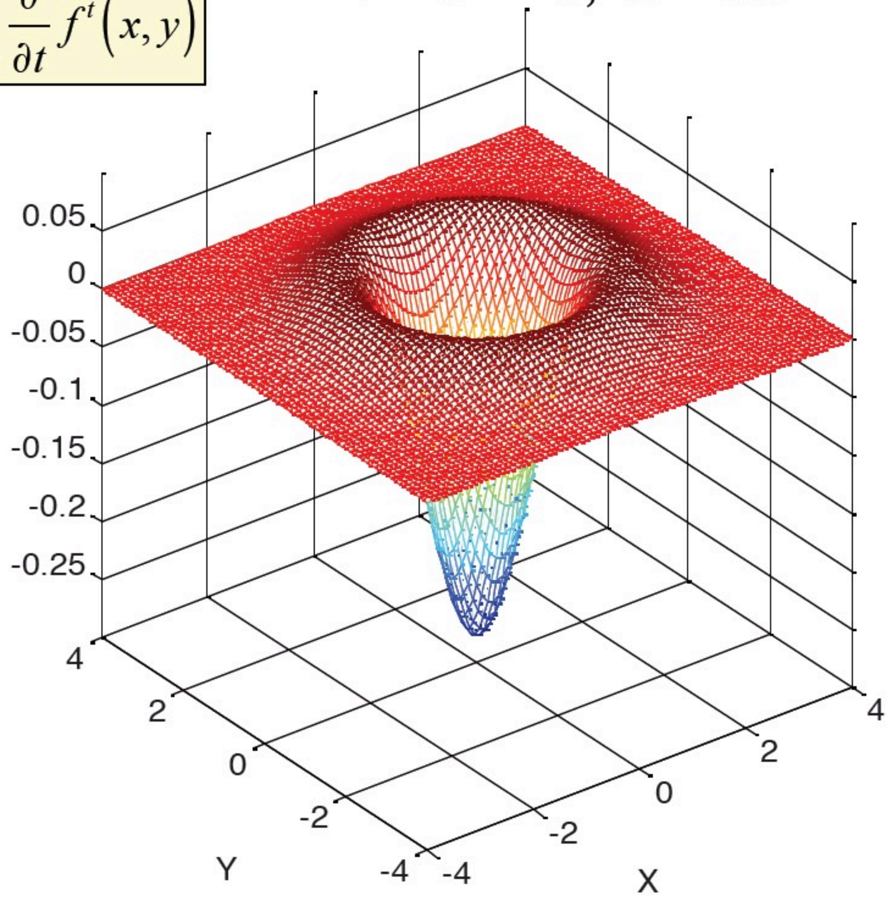
Difference of Gaussians

$t = \sigma^2 = 1$

$$\boxed{\frac{1}{2}\nabla^2 f^t(x,y) = \frac{\partial}{\partial t} f^t(x,y)}$$

$t = \sigma^2 = 1, \ k = 1.1$





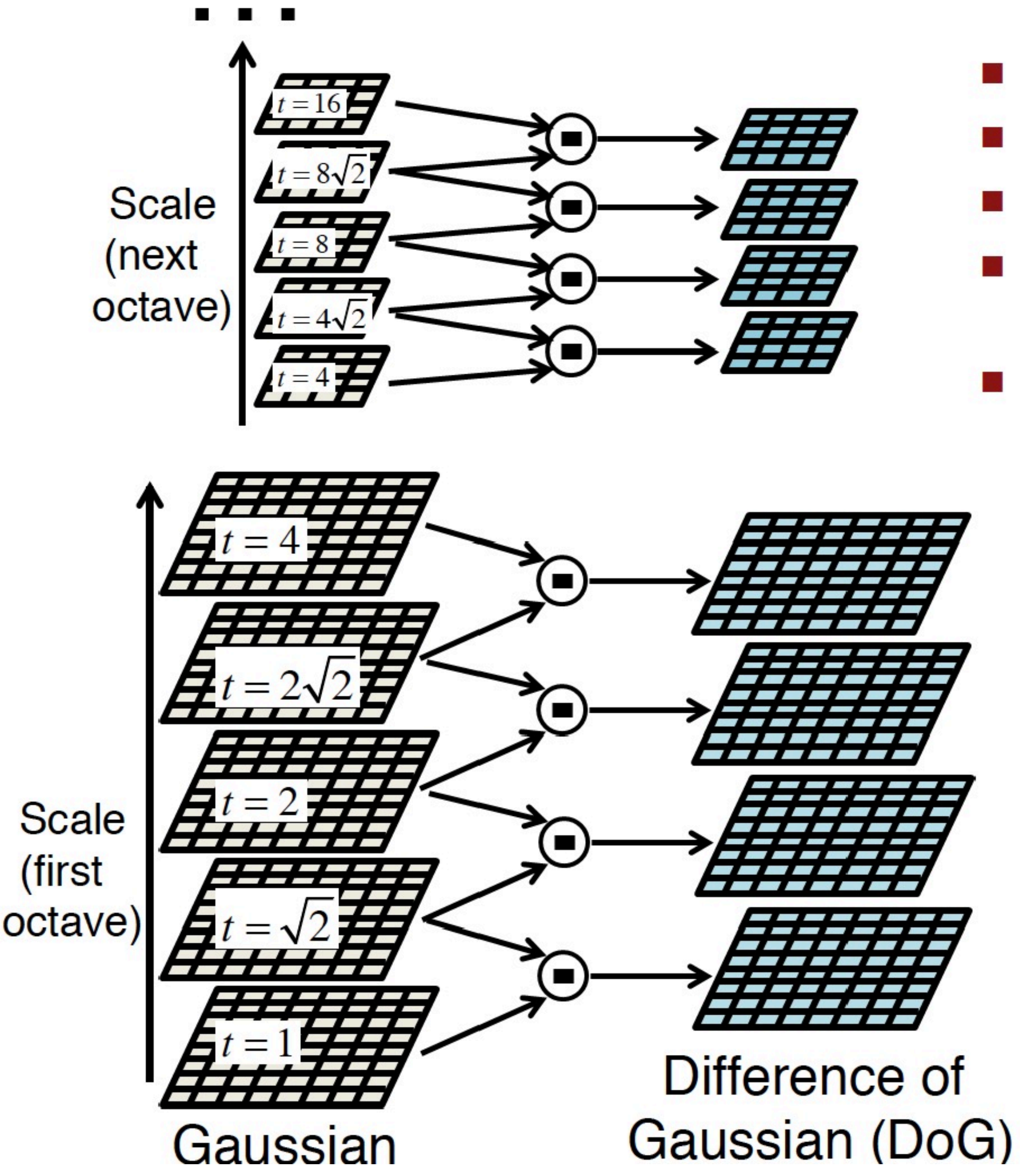
$$LoG(x,y) = -\frac{1}{\pi t^2}\left(1 - \frac{x^2+y^2}{2t}\right)e^{-\frac{x^2+y^2}{2t}}$$

$$DoG(x,y) = \frac{1}{(k-1)t}\left(g^{k^2 t}(x,y) - g^t(x,y)\right)$$

[courtesy B. Girod, (c) 2013 Stanford University]

From http://web.stanford.edu/class/ee368/Handouts/Lectures/2016_Autumn/14-ScaleSpace_16x9.pdf

Generating Laplacian Scale Space using Difference of Gaussian

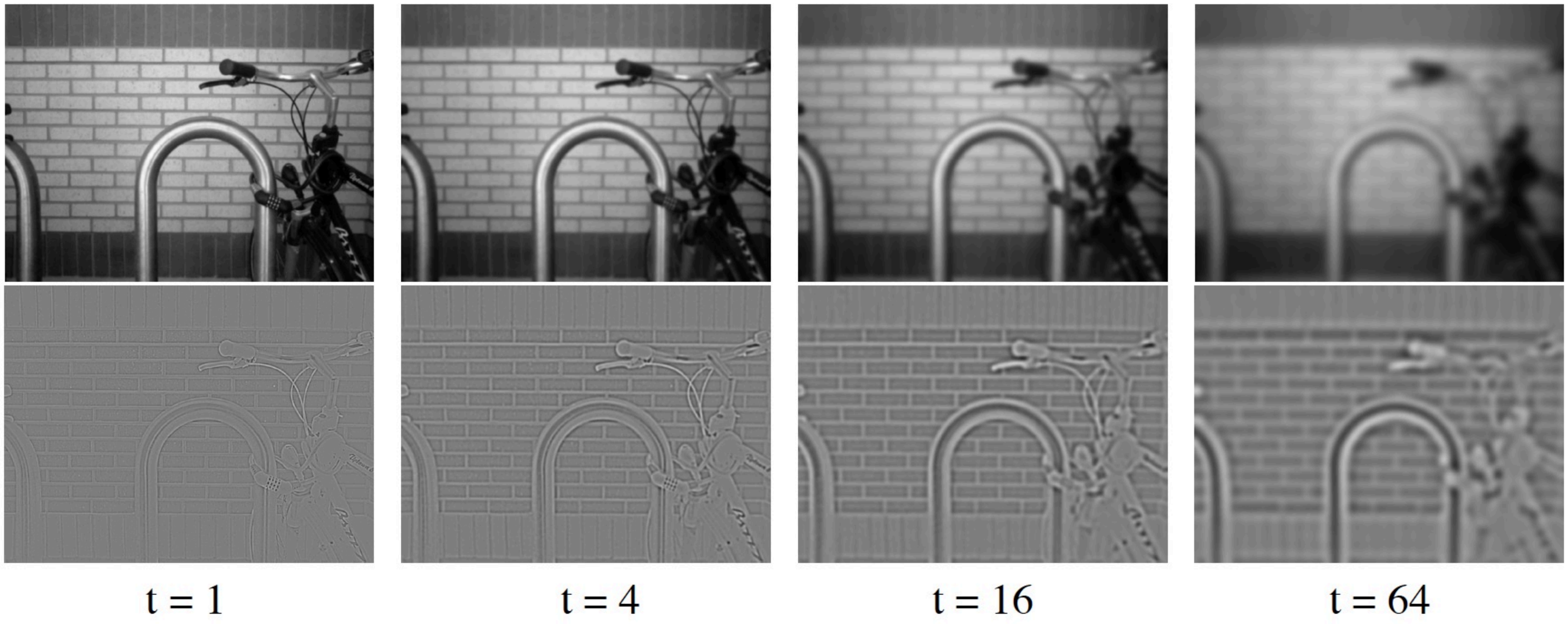


[courtesy B. Girod, (c) 2013 Stanford University]

Note: In this example, t represents σ^2 , k=2 ¼, and each octave (σ doubles) consists of 5 scales.
Lowe has found 3 scales (using k=sqrt(2)) is best (two LOG images at each scale)

From http://web.stanford.edu/class/ee368/Handouts/Lectures/2016_Autumn/14-ScaleSpace_16x9.pdf

Scale Space: Gaussian vs. Laplacian



[courtesy B. Girod, (c) 2013 Stanford University]

In the above notation: $t = \sigma^2, \quad \sigma = \sqrt{t}$

From http://web.stanford.edu/class/ee368/Handouts/Lectures/2016_Autumn/14-ScaleSpace_16x9.pdf

Pop Quiz

- You need to write a program to generate the Gaussian and Laplacian scale space, what would be the steps in the program?
- How is this different from the Laplacian Pyramid ?

Pop Quiz

- You need to write a program to generate the Gaussian and Laplacian scale space, what would be the steps in the program?
- Steps:
  - Generate Guassian filters with varying scales $\sigma_n=\sigma_0\, k^{n,}$
  - Convolve the image with Gaussian filters with $\sigma_{n.}$ Down sample everytime scale doubles -> Gaussian scale space
  - Take difference of two adjacent images in the same resolution -> Laplacian scale space
- How is this different from the Laplacian Pyramid ?
  - Laplacian pyramid image at level k = difference between image at level k – upsampled image from level k-1
  - Laplacian image at scale k = Gaussian image at scale k –Gaussian image at k-1
  - No explicit upsampling operation, directly approximate Laplacian filtering
  - Can have multiple scales at the same resolution

Finding Extrema in the Laplacian Scale Space

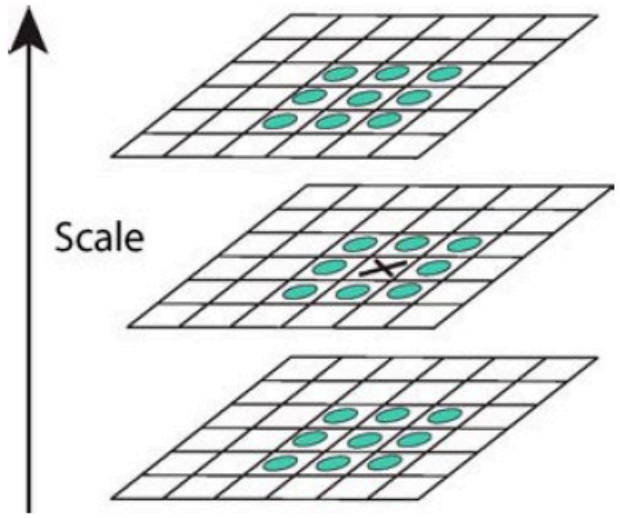
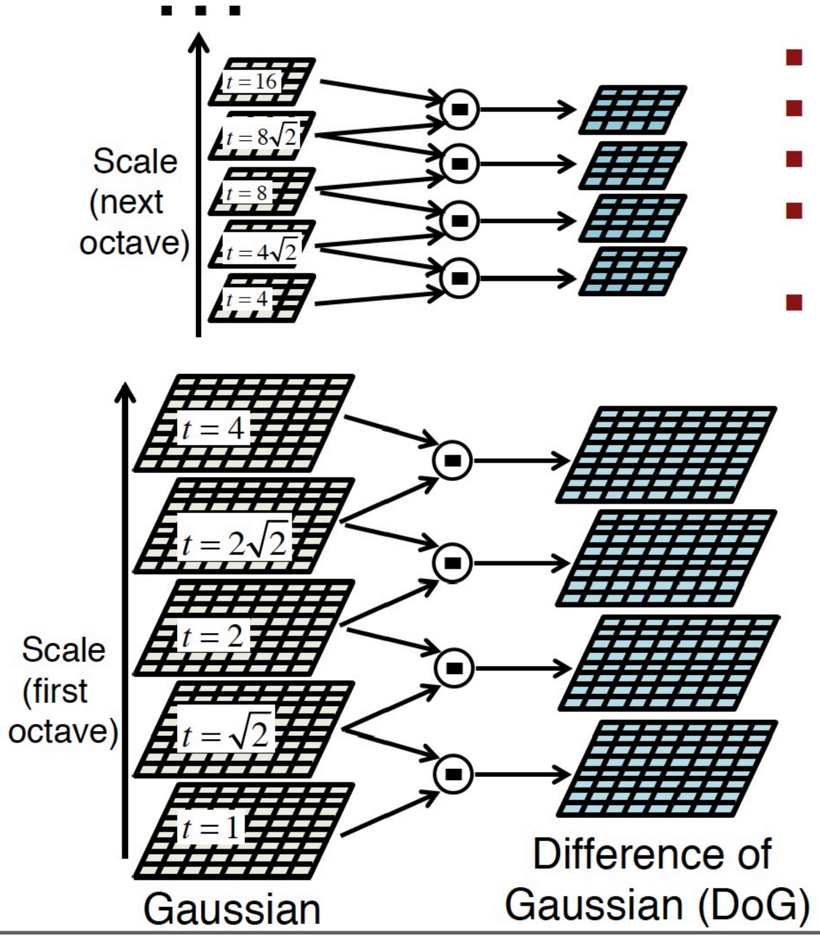


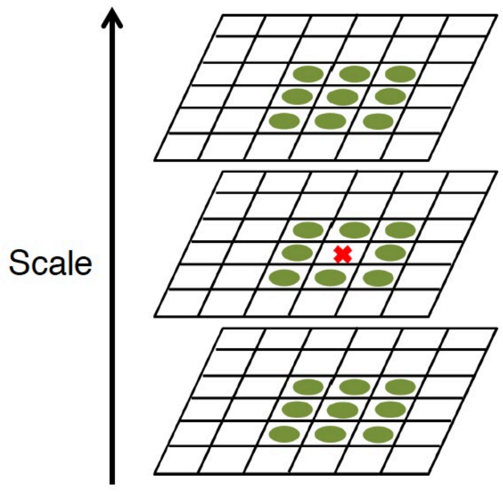
Figure 2. Maxima and minima of the difference-of-Gaussian images are detected by comparing a pixel (marked with X) to its 26 neighbors in 3×3 regions at the current and adjacent scales (marked with circles).

Initially locate the extrema (maximum or minimum) by comparing with 26 neighboring pixels. This position is then refined with a shift.

SIFT Feature Detector (Lowe 1999, 2004)



- SIFT - Scale-Invariant Feature Transform
- Decompose image into DoG scale-space representation
- Detect minima and maxima locally and across scales
- Fit 3-d quadratic function to localize extrema with sub-pixel/sub-scale accuracy *[Brown, Lowe, 2002]*
- Eliminate edge responses based on Hessian



[Lowe, 1999, 2004]

[courtesy B. Girod, (c) 2013 Stanford University]

From http://web.stanford.edu/class/ee368/Handouts/Lectures/2016_Autumn/14-ScaleSpace_16x9.pdf

Harris-Laplacian Detector

- Detect Harris corners at multiple scales (on the Gaussian filtered images at scales σ_n , n=0,1,…,)
- For each detected Harris corner x_h, y_h at any scale, determine the characteristic scale (the scale at which Laplacian/DoG achieves maximum or minimum)

$$\sigma_h = \operatorname{argmax} \left| \sigma^2 \nabla L(x_h, y_h, \sigma) \right|$$

- Keep the point at x_h, y_h, σ_h

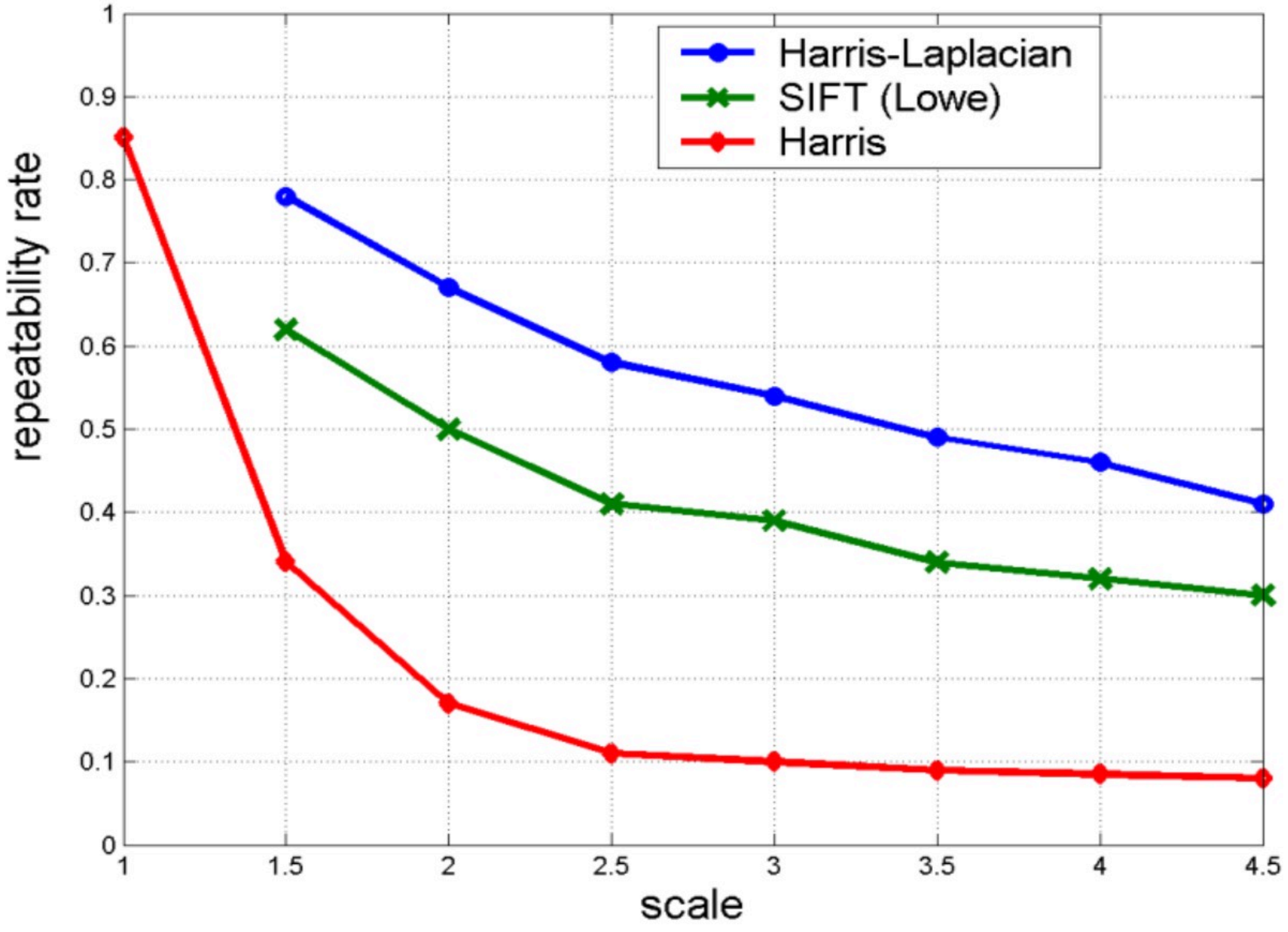
- Ref: K. Mikolajczyk and C. Schmid, “Indexing Based on Scale Invariant Interest Points,” Proc. Eighth Int’l Conf. Computer Vision, pp. 525-531, 2001.

Comparison of Different Feature Detectors

- Evaluation criterion:
  - Repeatability under scaling, rotation, view angle change (affine transform), and noise

- Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., and Van Gool, L. J. (2005). A comparison of affine region detectors. International Journal of Computer Vision , 65(1-2):43–72.
  - Code for different detectors: http://www.robots.ox.ac.uk/vgg/research/affine/ .
  - Multiscale Harris and Hessian are better than SIFT detector.
  - Hessian slightly better than Harris.

Comparison of Feature Detectors



[courtesy B. Girod, (c) 2013 Stanford University]

From http://web.stanford.edu/class/ee368/Handouts/Lectures/2016_Autumn/14-ScaleSpace_16x9.pdf

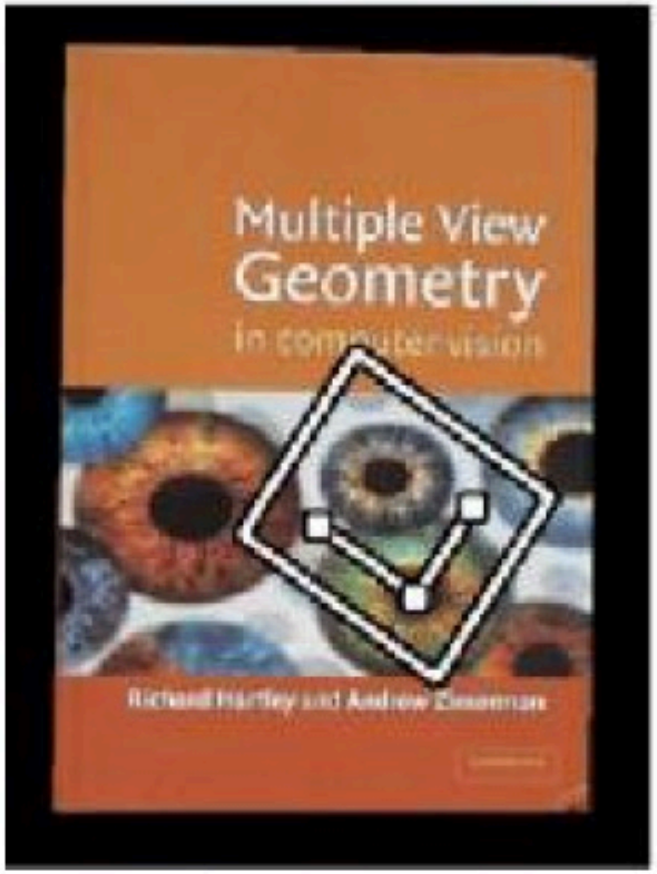
Lecture Outline

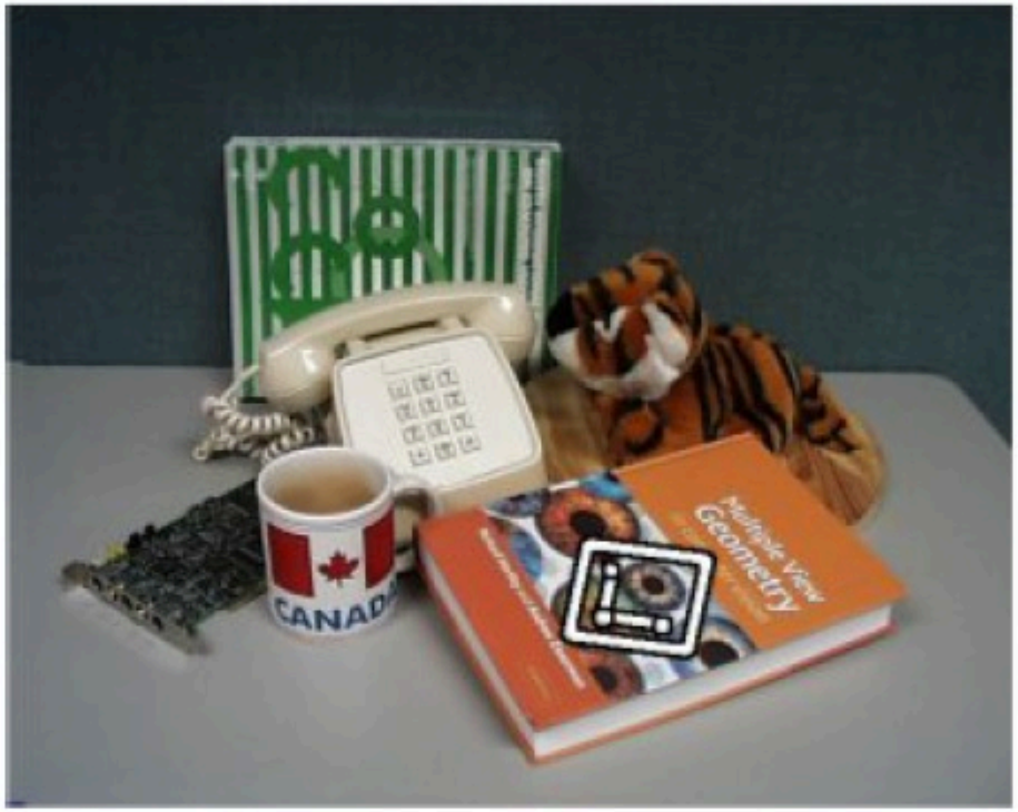
- Need for Features and Feature Descriptors
- Feature Detectors
  - Desired properties of features
  - Harris Detector
  - Scale Space
  - Harris-Laplace detector
  - SIFT detector
  - Scale and orientation detection
- ⇨ Feature descriptors
  - SIFT
  - SURF
- Deep learning for feature detection and description
- Image classification using Bag of Visual Words

Feature Descriptor

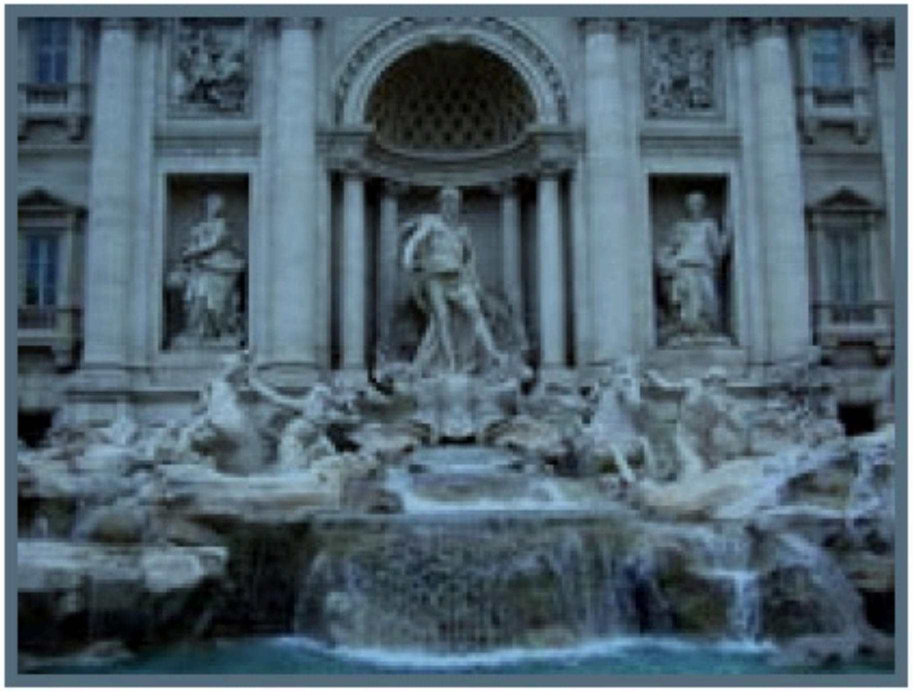
- Why do we need them?
  - To find corresponding features between two images
- Simple approach
  - Using the image values surrounding a feature point as its descriptor
  - Compare two descriptors using Euclidean distance or other norms
  - Not robust to changes in view angle/distance and positions and sensitive to occlusion and lighting/contrast changes

Need for rotation invariance

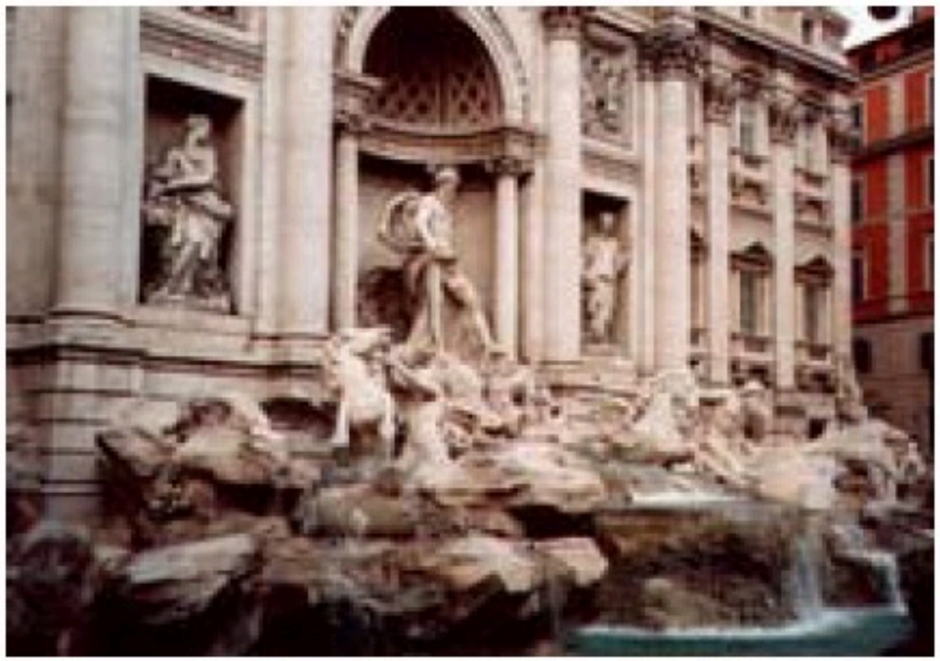




From https://courses.cs.washington.edu/courses/cse455/08wi/lectures/features.pdf



by Diva Sian

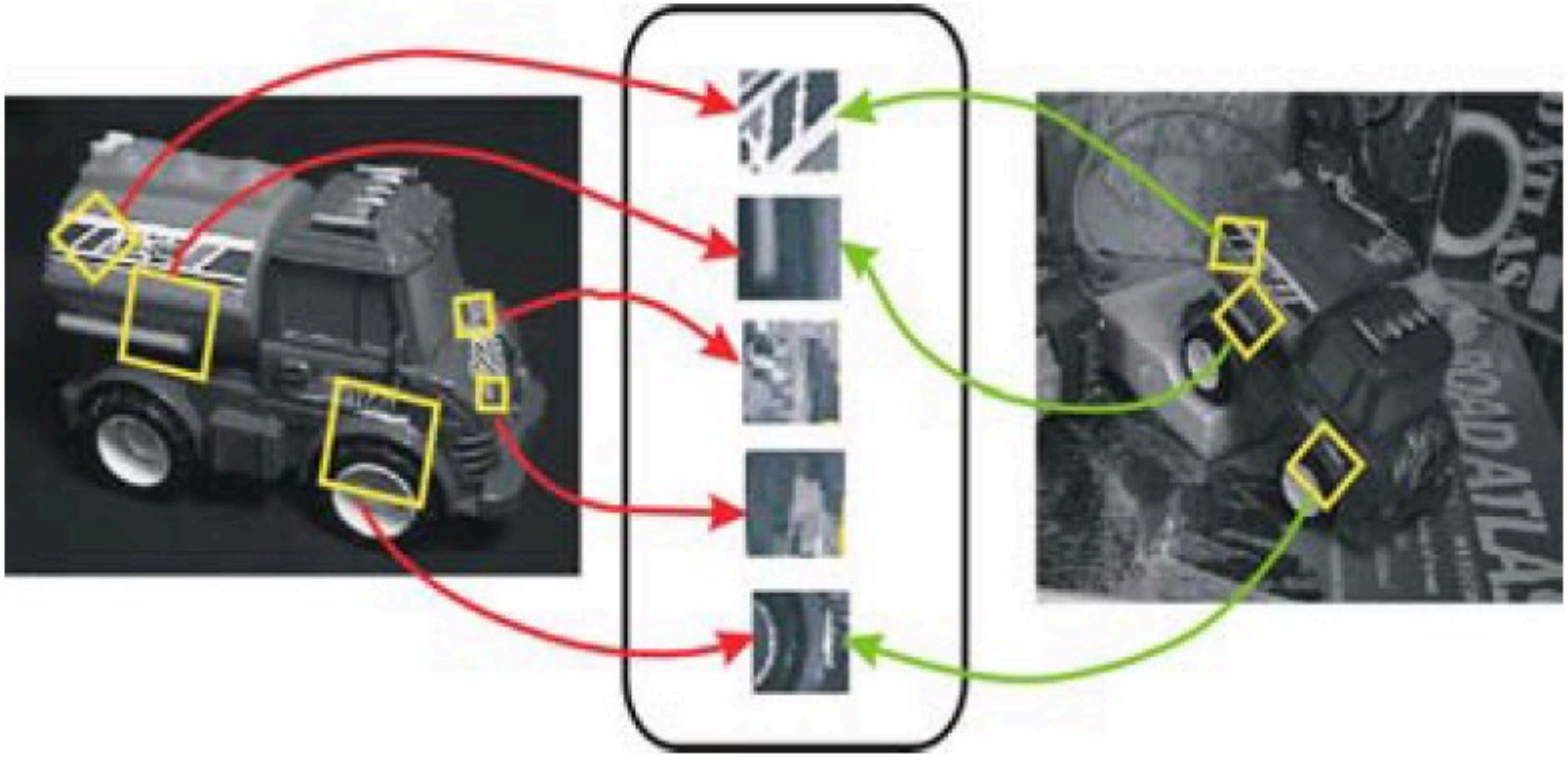


by scgbt

From https://courses.cs.washington.edu/courses/cse455/08wi/lectures/features.pdf

General Approach for Feature Descriptors

- Desirable properties
  - Invariant to shift, scale and rotations and contrast differences
- General approach:
  - Determine the scale where a feature is most stable
  - Determine the orientation of the small patch surrounding this feature point at this scale
  - Characterize a small patch tilted in this orientation and scale with a descriptor
  - The descriptor should be invariant to intensity shift or scaling, and small position shift
  - Most popular: SIFT descriptor by David Lowe (UBC)
    - http://www.cs.ubc.ca/~lowe/keypoints/
    - Caveat: SIFT feature detector vs. SIFT descriptor!



Feature Descriptors

From https://courses.cs.washington.edu/courses/cse455/08wi/lectures/features.pdf

Descriptor use normalized patches where the dominant line in an original patch is rotated to a standard direction (e.g. vertical!) and the original patch is scaled to the same size!

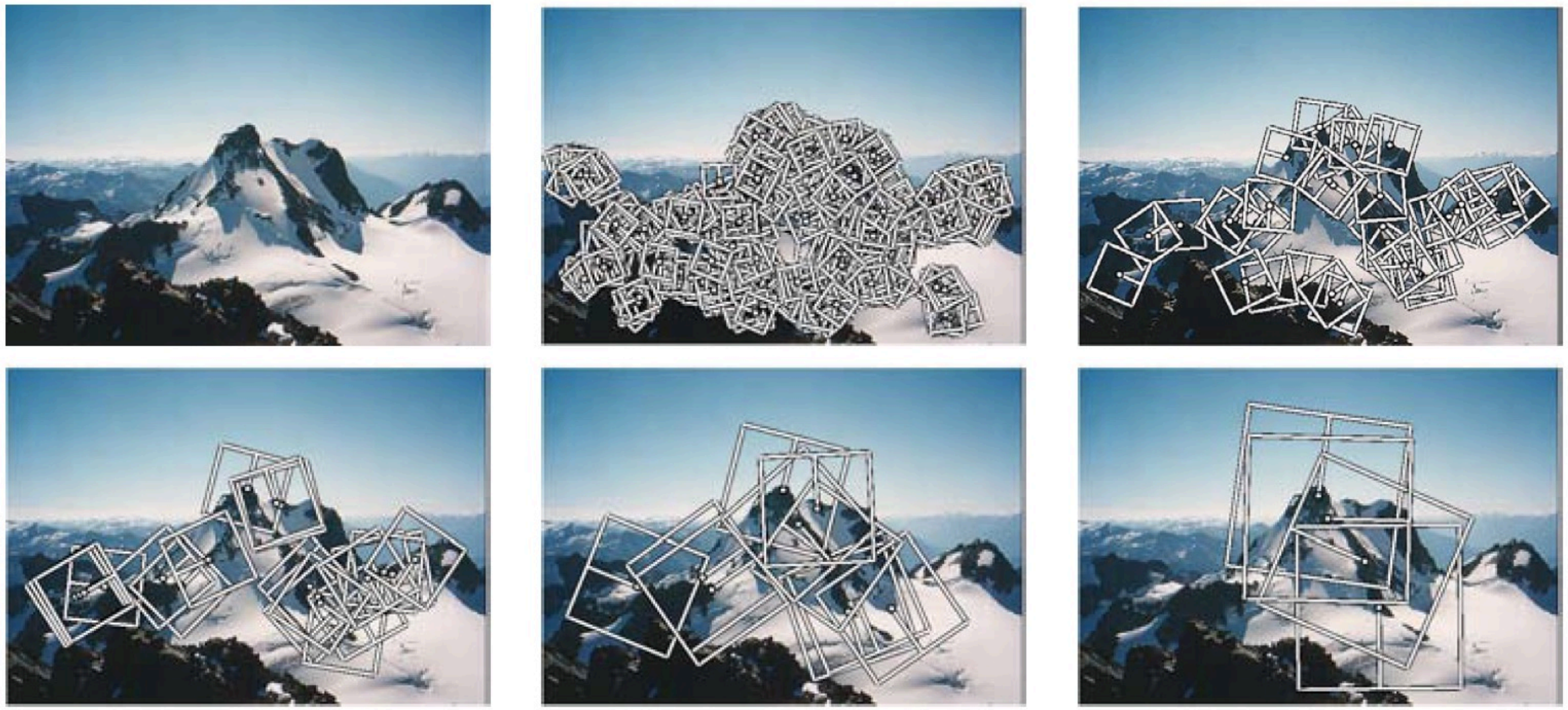
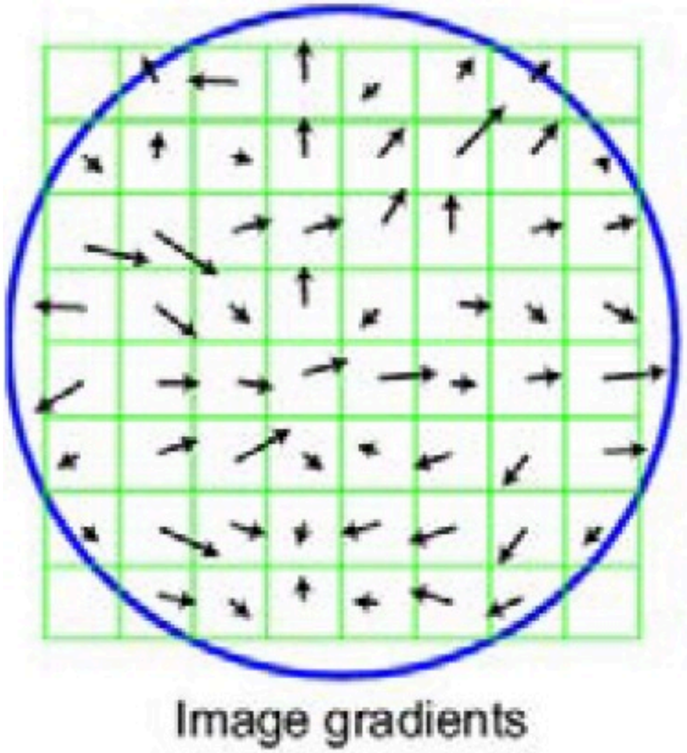


Figure 4.10 Multi-scale oriented patches (MOPS) extracted at five pyramid levels (Brown, Szeliski, and Winder 2005) © 2005 IEEE. The boxes show the feature orientation and the region from which the descriptor vectors are sampled.

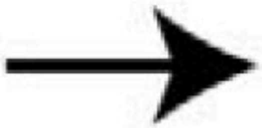
A descriptor is generated for each feature point detected at a particular scale by extracting a patch surrounding the point based on its scale and orientation.

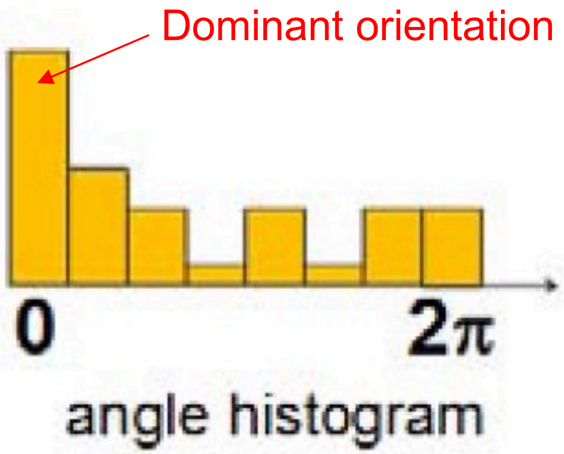
From Szeliski, *Computer Vision: Algorithms and Applications*, 2010

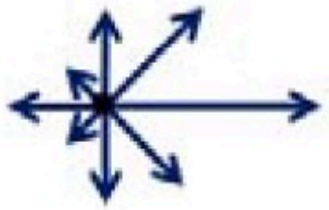
Determination of Dominant Orientation



Sum of gradient magnitude with each orientation







Another representation of the angle histogram

From Szeliski, *Computer Vision: Algorithms and Applications*, 2010

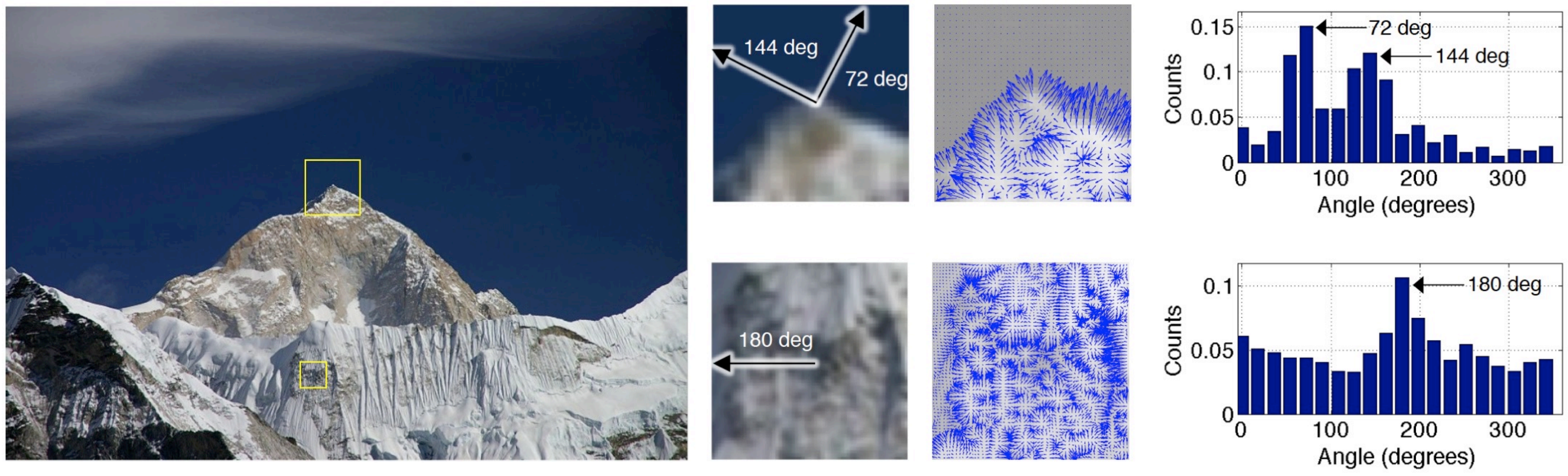
Histogram of Orientation of Gradient (HoG)

- Extract a patch surrounding the detected feature point at the detected scale
- Derive image gradient I_x and I_y and determine the magnitude and orientation for every pixel. Zero out weak edges (set small magnitude to 0)
  - Using Derivative of Gaussian filters with the same σ as the detected scale,
  - Or using central difference on the Gaussian filtered image at that scale

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

- Quantize the orientation to a chosen set of bins in (0, 2π).
- Apply Gaussian weighting at a larger σ (σ=1/2 width of patch size) to the gradient magnitude
- Generate a histogram, where the entry for each orientation bin is the sum of weighted gradient magnitude of all pixels with that orientation
- Finding the orientation with the peak frequency
- For more details see [Lowe2004]

HoG Examples

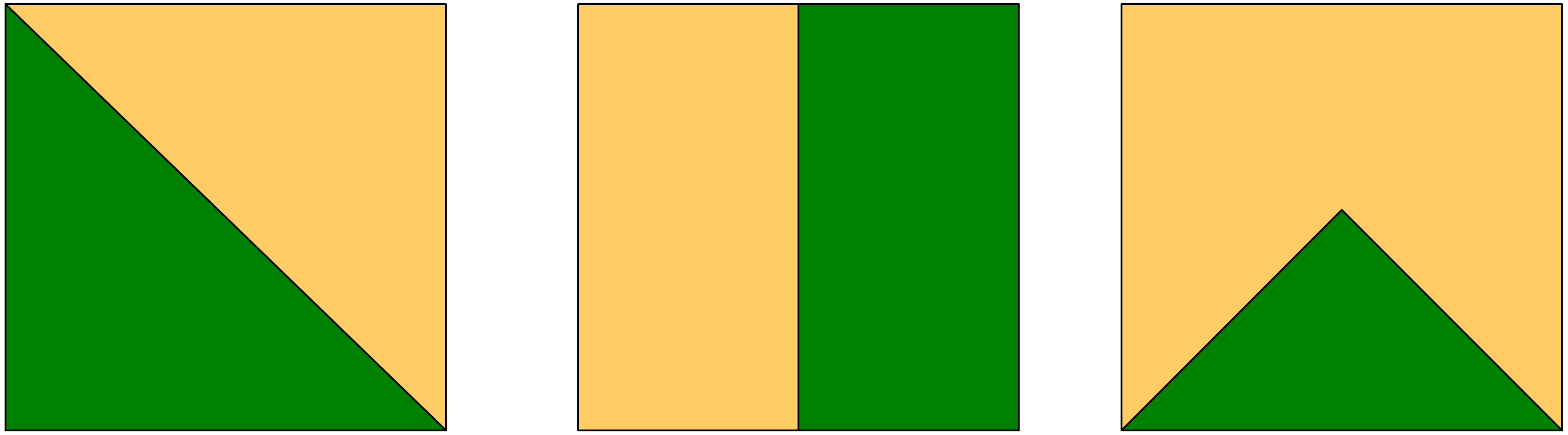


[courtesy B. Girod, (c) 2013 Stanford University]

From: http://web.stanford.edu/class/ee368/Handouts/Lectures/2016_Autumn/15-Featurebased_Image_Matching_16x9.pdf

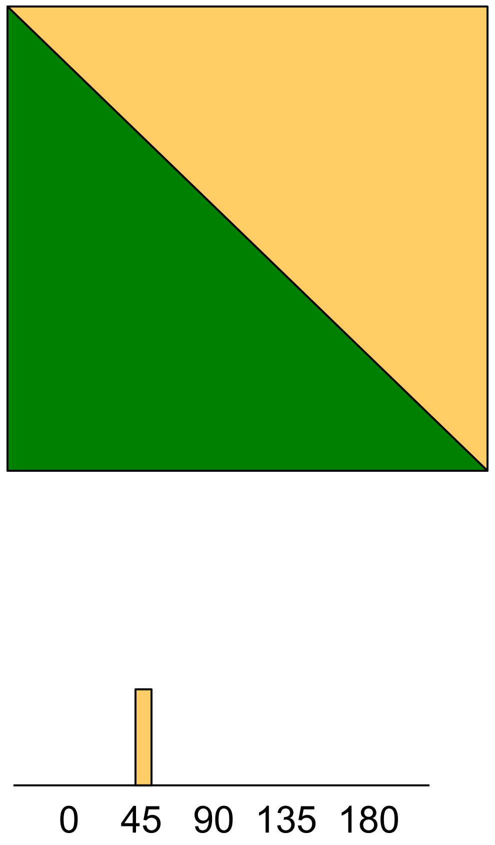
Pop Quiz

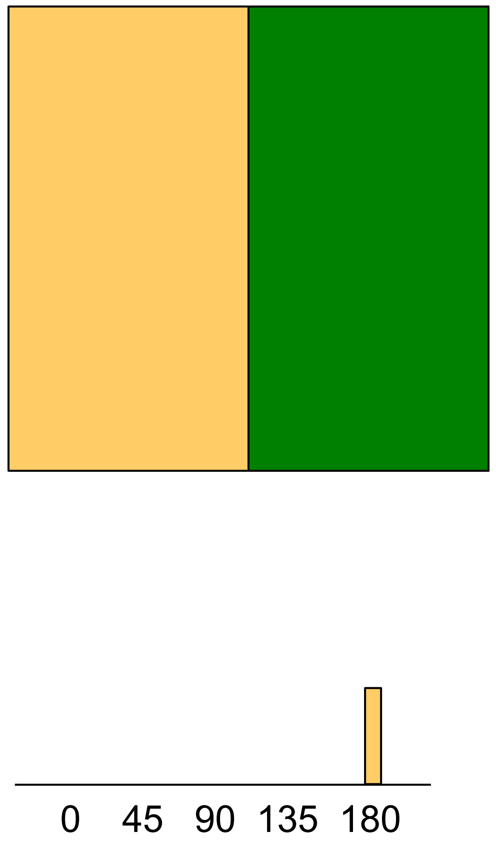
- What will the HoG look like for following patches?
  - Assume yellow has a higher brightness than green

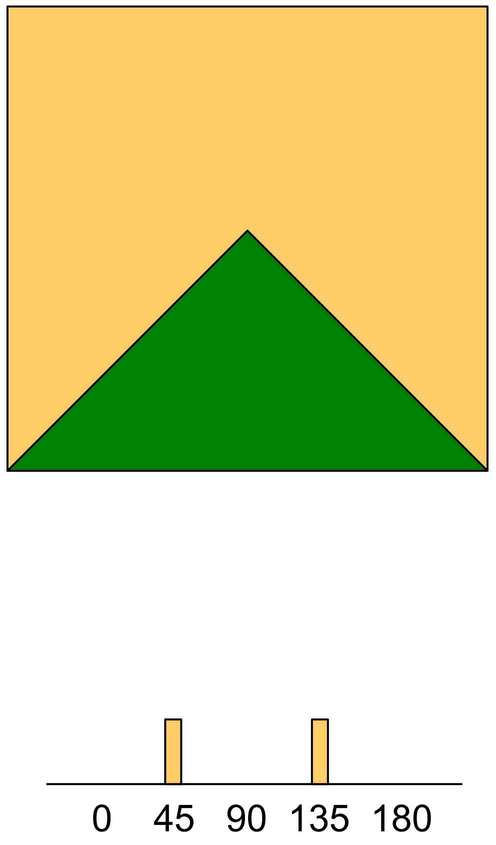


Pop Quiz

- What will the HoG look like for following patches?
  - Assume yellow has a higher brightness than green







SIFT Descriptor For a Given Patch (Basic Idea)

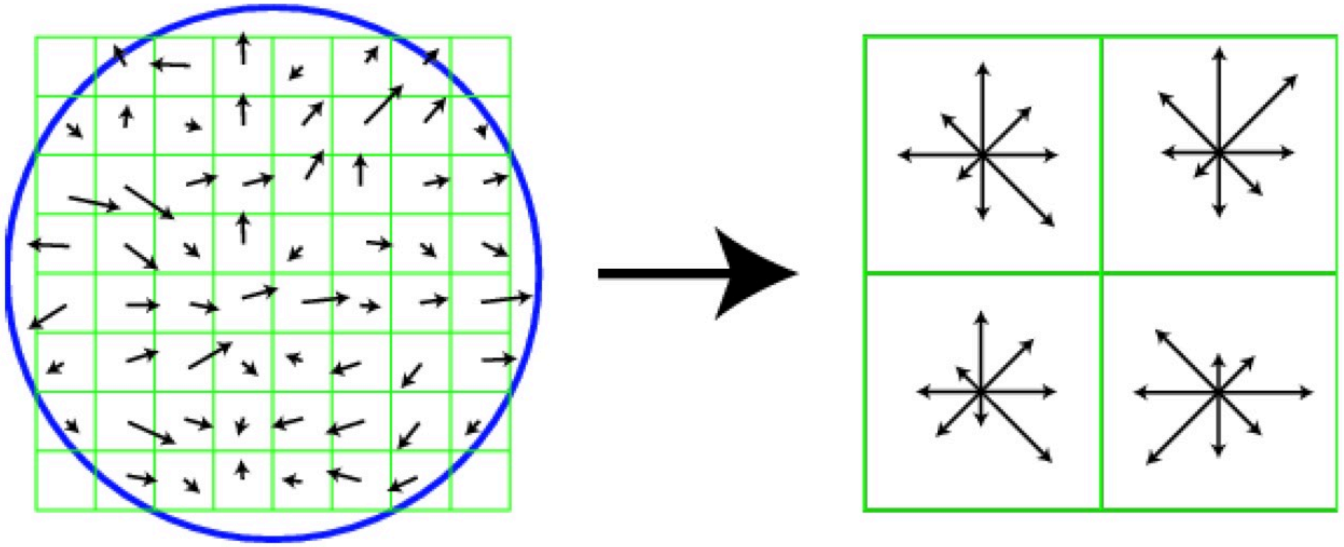
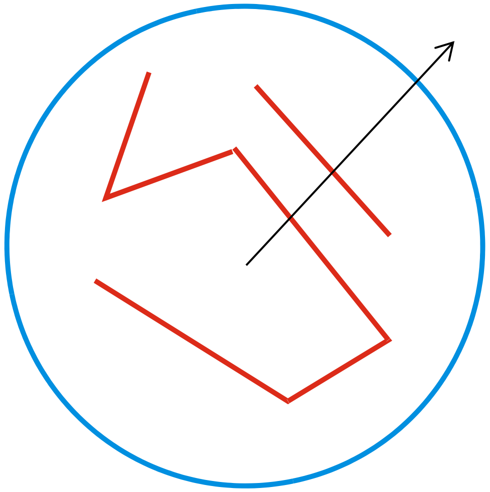


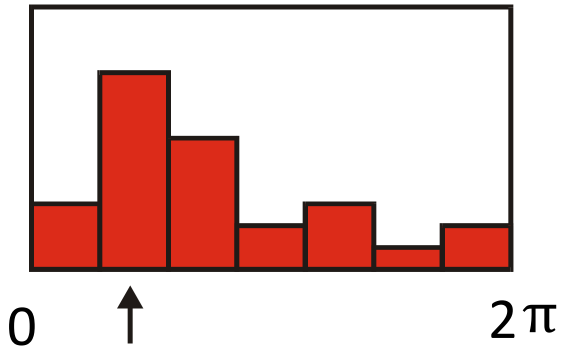
Figure 4.18 A schematic representation of Lowe's (2004) scale invariant feature transform (SIFT): (a) Gradient orientations and magnitudes are computed at each pixel and weighted by a Gaussian fall-off function (blue circle). (b) A weighted gradient orientation histogram is then computed in each subregion, using trilinear interpolation. While this figure shows an 8×8 pixel patch and a 2×2 descriptor array, Lowe's actual implementation uses 16×16 patches and a 4×4 array of eight-bin histograms.

From Szeliski, *Computer Vision: Algorithms and Applications*, 2010

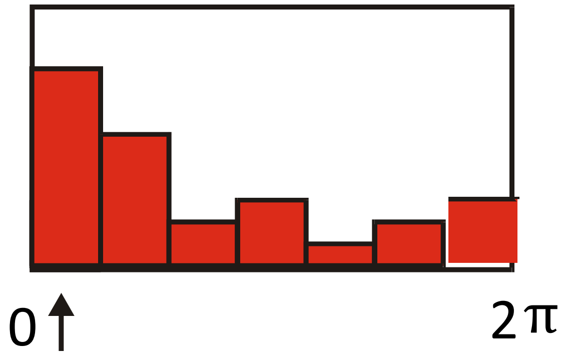
Orientation invariance:
Shift of HoG to Align with Reference Orientation



Patch near a feature point



Original HoG



Shifted HoG

Adapted from https://courses.cs.washington.edu/courses/cse455/13au/: Lecture on "interest points"

How to deal with feature points with different scales?

- Generate patches with a fixed size at the detected scale in the Gaussian scale space
- The image is halved for every increasing octave
- The effective patch size doubles for every increasing octave!

SIFT Descriptor

- Put a patch of 16x16 around each feature point at the detected scale in the Gaussian scale space.
- Generate a HOG for the entire patch, determine the dominant orientation
- Divide it into 4x4 cells, each cell 4x4 in size
- Generate a (HoG) for each cell with 8 bins
- Shift each HoG so that the dominant orientation is in the first bin (0 degree).
- Concatenate the shifted HoG of each cell into a single descriptor
  - 16 cells * 8 orientations= 128 dimension descriptor!
- Normalize the descriptor so that it is invariant to intensity variation
  - Set all entries >0.2 to 0.2
  - Normalize the resulting descriptor
  - Robust to contrast/brightness variation!
- SIFT descriptor can be used for feature points extracted by other methods (e.g. Harris).
  - Harris-Laplacian Detector + SIFT Descriptor is a popular combination!

Pop Quiz

- Why use the HoG to describe a patch?
- Why do you want to shift the HoG based on the dominant direction
- Why use scale space?

Pop Quiz

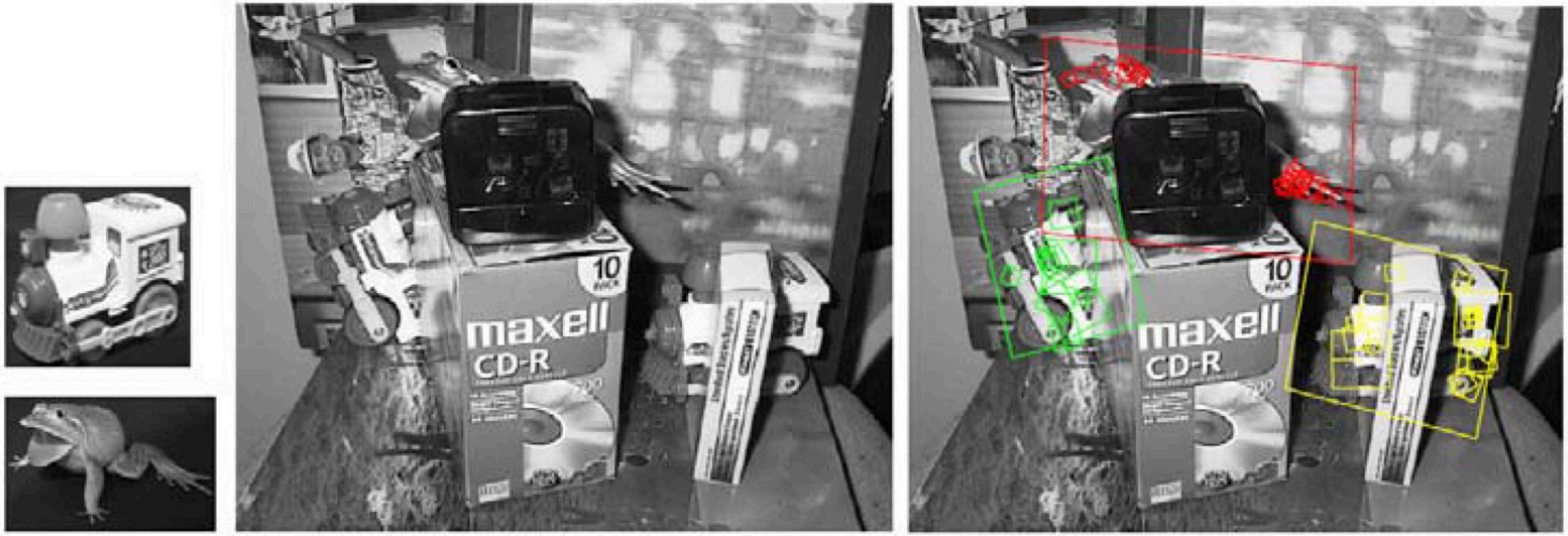
- Why use the HoG to describe a patch?
  - Invariant to brightness and position change
- Why do you want to shift the HoG based on the dominant direction
  - Invariant to rotation
- Why use scale space?
  - To achieve scale invariance

Power of SIFT Descriptor

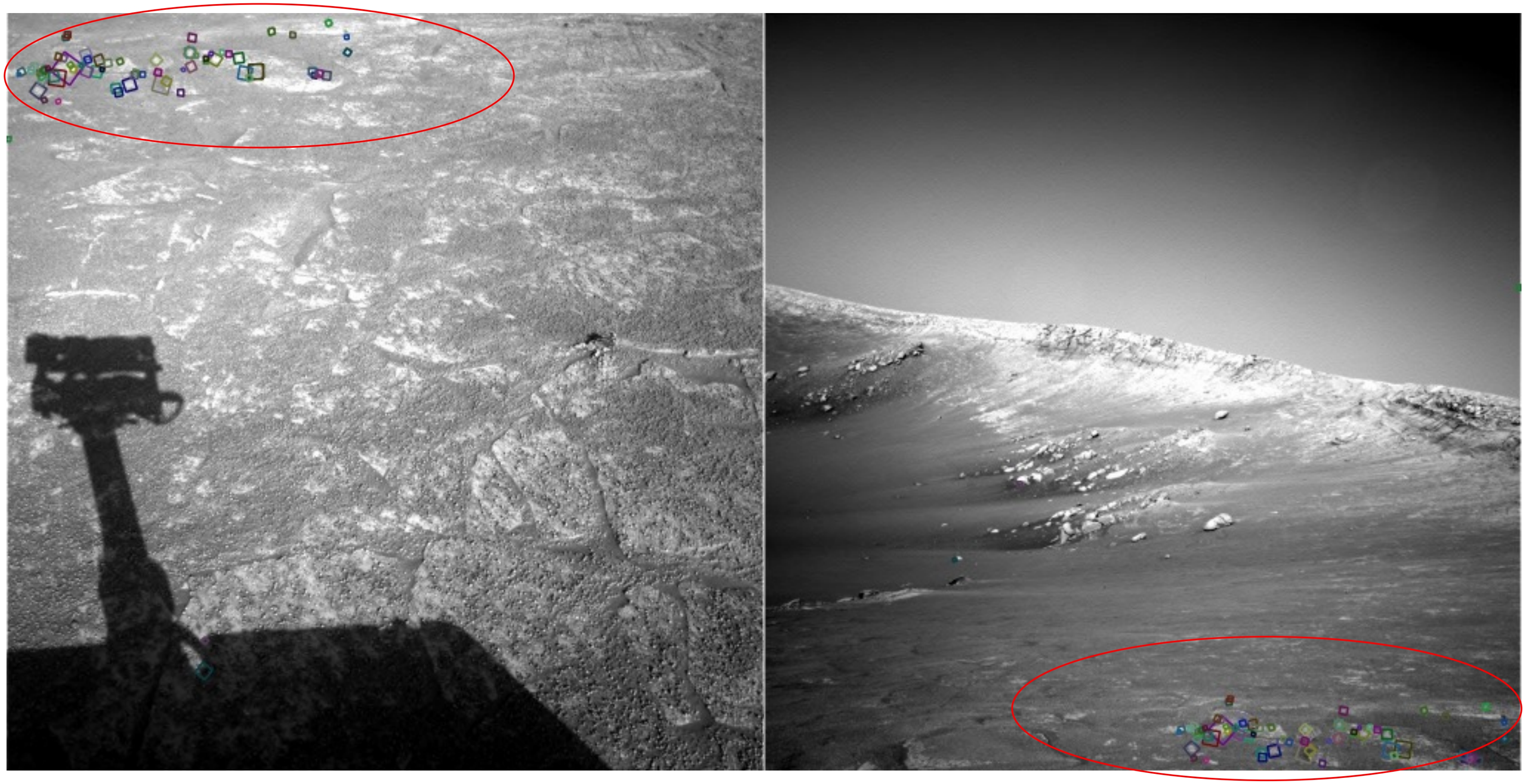
- Extraordinarily robust matching technique
- Can handle changes in viewpoint
  - Up to about 60 degree out of plane rotation
- Can handle significant changes in illumination
  - Sometimes even day vs. night (below)
- Fast and efficient—can run in real time
  - Lots of code available
  - http://people.csail.mit.edu/albert/ladypack/wiki/index.php/Known_implementations_of_SIFT
  - http://www.cs.ubc.ca/~lowe/keypoints/

From https://courses.cs.washington.edu/courses/cse455/16wi/notes/index.html, lecture on descriptors

Can you find the toy car and frog in the picture?



From Szeliski, *Computer Vision: Algorithms and Applications*, 2010



NASA Mars Rover images, Figure by Noah Snavely

From https://courses.cs.washington.edu/courses/cse455/16wi/notes/index.html, lecture on descriptor

Other Descriptors

- SURF (Speed-Up Robust Feature) (can be computed very fast)
- 3D SIFT
- 3D SURF
- GLOH (Gradient Location and Orientation Histogram)

- Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence , 27(10):1615–1630.
  - GLOH is best, closely followed by SURF

Other kinds of descriptors

- There are descriptors for other purposes
  - Describing shapes
  - Describing textures
  - Describing features for image classification (e.g. bag of words, to be covered later)

From https://courses.cs.washington.edu/courses/cse455/16wi/notes/index.html, lecture on descriptor

Lecture Outline

- Need for Features and Feature Descriptors
- Feature Detectors
  - Desired properties of features
  - Harris Detector
  - Scale Space
  - Harris-Laplace detector
  - SIFT detector
  - Scale and orientation detection
- Feature descriptors
  - SIFT
  - SURF
- ⇨ Deep learning for feature detection and description
- Image classification using Bag of Visual Words

Learnt Feature Detectors and Descriptors (not required)

- Recent development in deep learning for feature detection and description
  - Yi, K. M., Trulls, E., Lepetit, V., and Fua, P. (2016). LIFT: Learned invariant feature transform. In European Conference on Computer Vision, pp. 467–483.
  - DeTone, D., Malisiewicz, T., and Rabinovich, A. (2018). SuperPoint: Selfsupervised interest point detection and description. In IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 224–236.
  - Balntas, V., Lenc, K., Vedaldi, A., Tuytelaars, T., Matas, J., and Mikolajczyk, K. (2019). HPatches: A benchmark and evaluation of handcrafted and learned local descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence, early access. (A good review of various approaches)

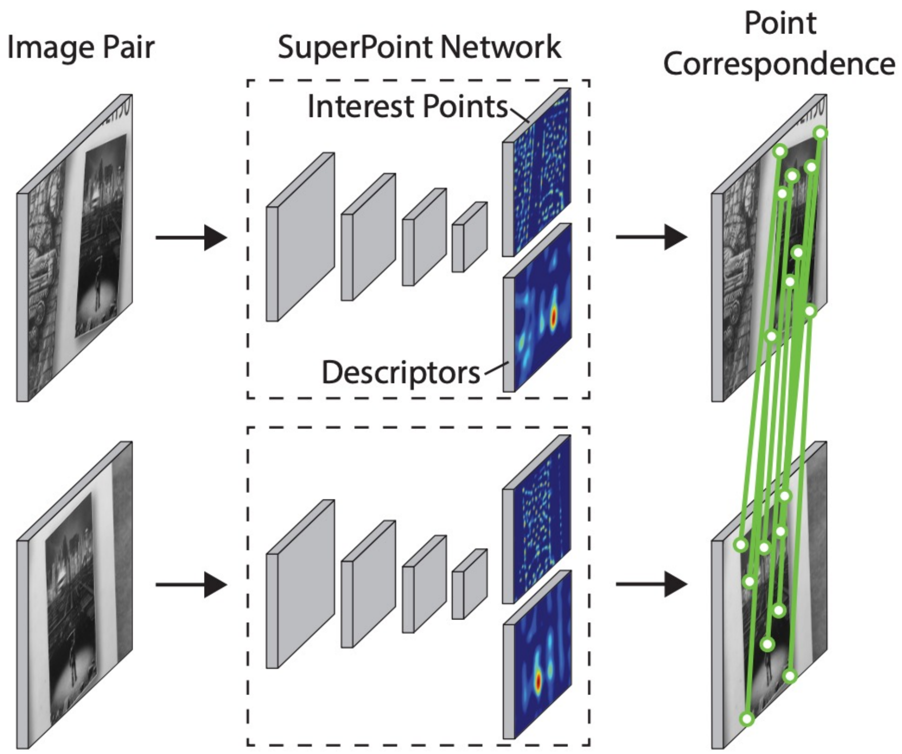


Figure 1. **SuperPoint for Geometric Correspondences.** We present a fully-convolutional neural network that computes SIFT-like 2D interest point locations and descriptors in a single forward pass and runs at 70 FPS on 480×640 images with a Titan X GPU.

Figure from: DeTone, D., Malisiewicz, T., and Rabinovich, A. (2018). SuperPoint: Selfsupervised interest point detection and description. In IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 224–236.

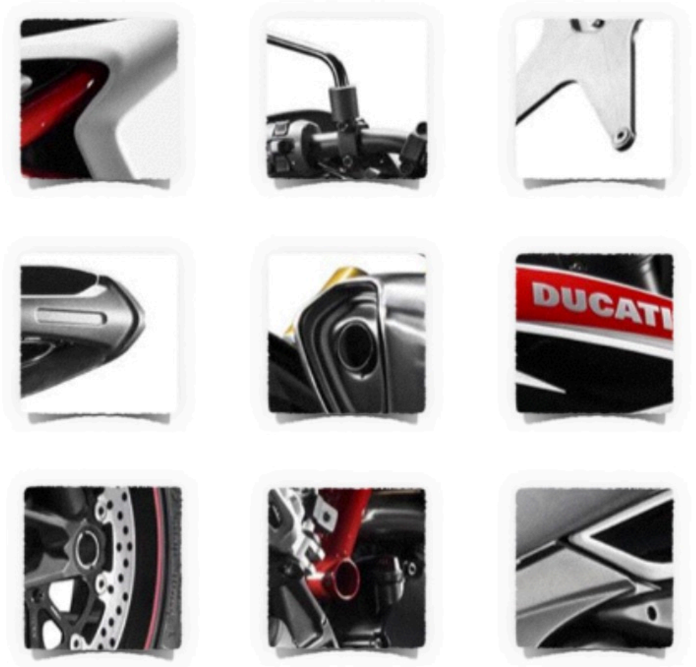
Lecture Outline

- Need for Features and Feature Descriptors
- Feature Detectors
  - Desired properties of features
  - Harris Detector
  - Scale Space
  - Harris-Laplace detector
  - SIFT detector
  - Scale and orientation detection
- Feature descriptors
  - SIFT
  - SURF
- Deep learning for feature detection and description
- ⇨ Image classification using Bag of Visual Words

Bag of Visual Words for Image Classification

Some local feature are very informative

An object as



a collection of local features

(bag-of-features)

- deals well with occlusion
- scale invariant
- rotation invariant

From http://www.cs.cmu.edu/~16385/lectures/Lecture12.pdf

BoW Pipeline for Image Classification

1. Extract features
2. Learn “visual vocabulary”
3. Quantize features using visual vocabulary
4. Represent images by frequencies of “visual words”

From http://www.cs.cmu.edu/~16385/lectures/Lecture12.pdf

1. **Extract features**

   At every detected feature points generate SIFT, HOG, or other descriptors

2. Learn "visual vocabulary"

3. Quantize features using visual vocabulary

4. Represent images by frequencies of "visual words"

From http://www.cs.cmu.edu/~16385/l

1. Extract features

2. **Learn "visual vocabulary"**

How to deduce these words ?

Group all features in the training set into a finite number of clusters (K-means clustering, GMM clustering, etc.), and represent each cluster by a "mean feature vector" (a visual word)

3. Quantize features using visual vocabulary

4. Represent images by frequencies of "visual words"

From http://www.cs.cmu.edu/~16385/lectures/Lecture12.pdf

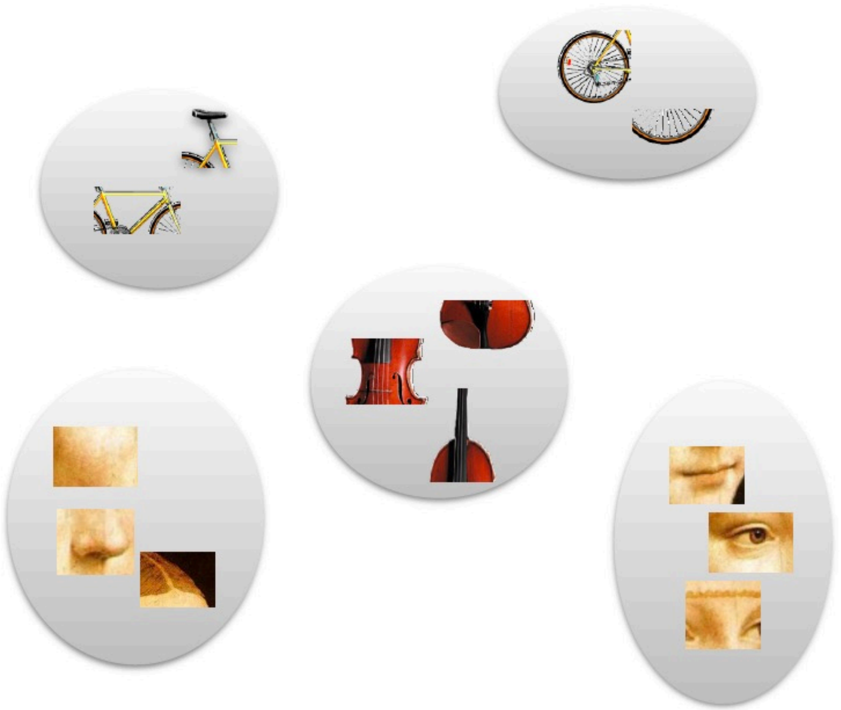
1. Extract features

2. Learn "visual vocabulary"

3. **Quantize features using visual vocabulary**

4. Represent images by frequencies of "visual words"

For a new image, replace the descriptor of each feature point by the "word" that is closest to the original descriptor (nearest neighbor search)



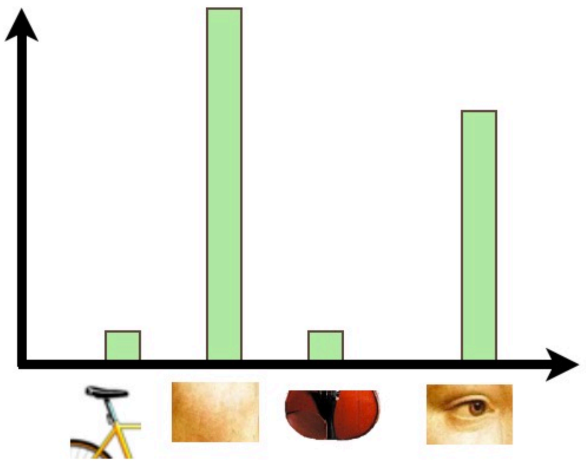
From http://www.cs.cmu.edu/~16385/lectures/Lecture12.pdf

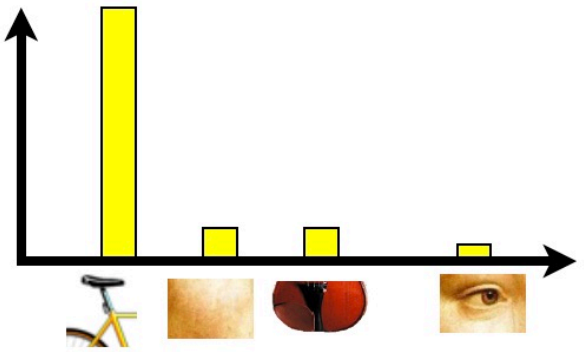
1. Extract features

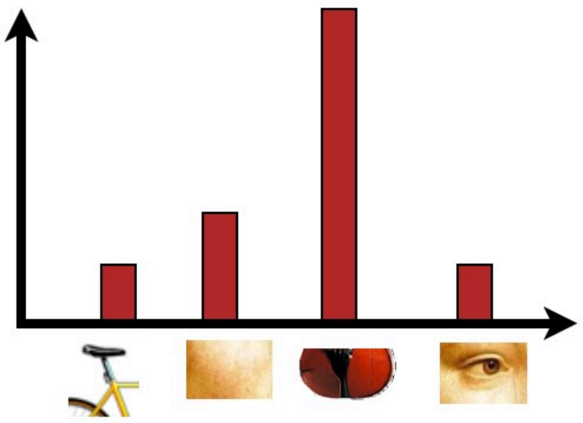
2. Learn “visual vocabulary”

3. Quantize features using visual vocabulary

4. **Represent images by frequencies of “visual words”**





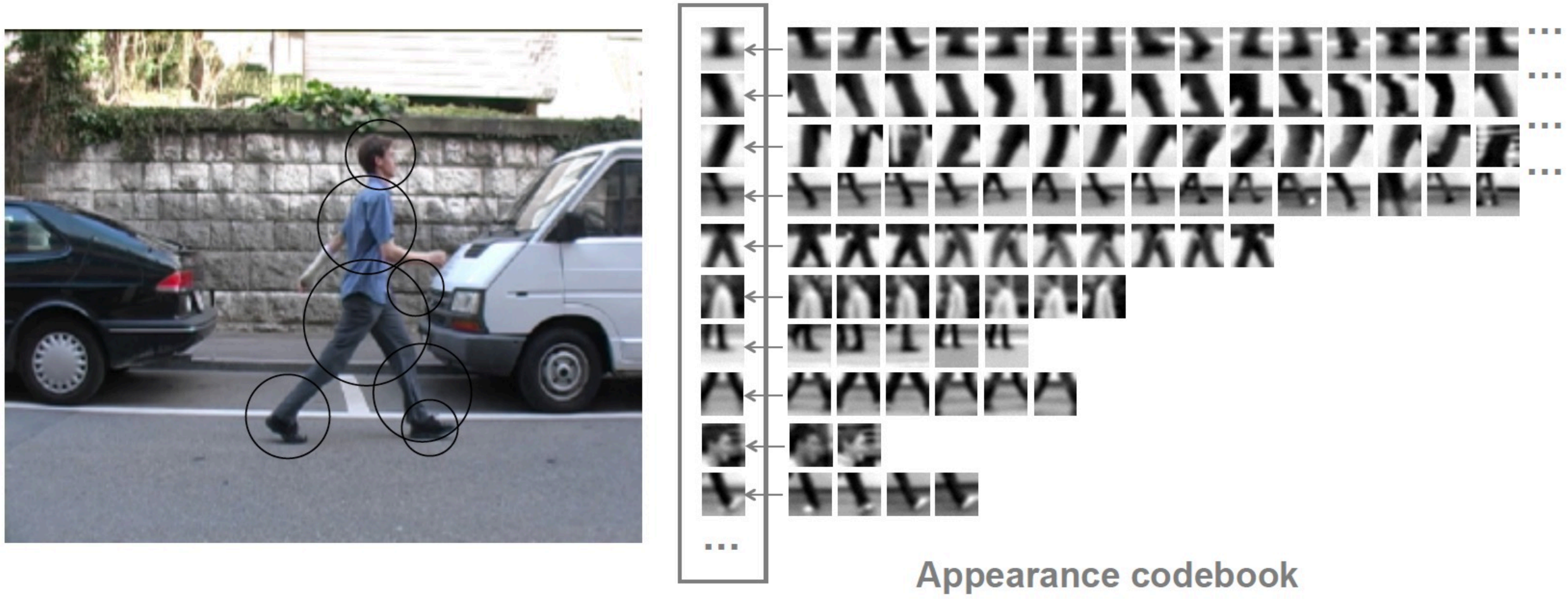


From
http://www.cs.cmu.edu/~16385/l

How to Learn Visual Dictionary?

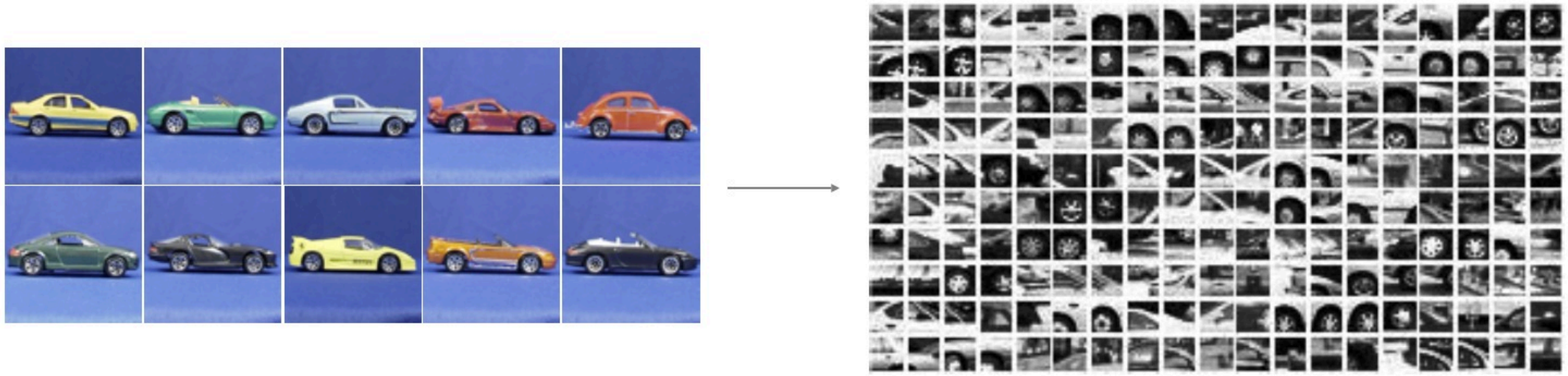
- Assume you have feature vectors extracted from many training images
- Apply K-means or other clustering algorithms to all training feature vectors!
- The training images should encompass categories to be classified
- How to select initial centroids?
  - K-means++
- How to determine K?
- Belong to the problem of “unsupervised learning”. Beyond the scope of this class. Self study!

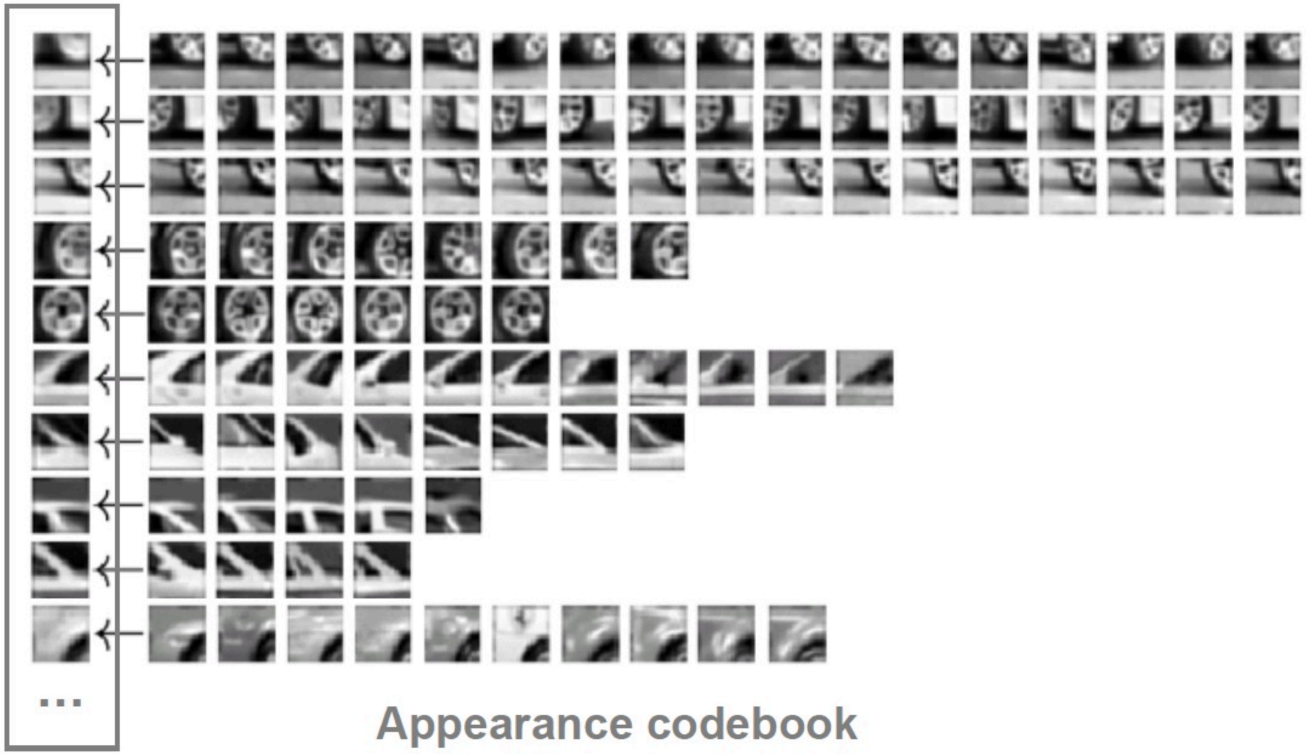
“Words” from people



From http://www.cs.cmu.edu/~16385/lectures/Lecture12.pdf

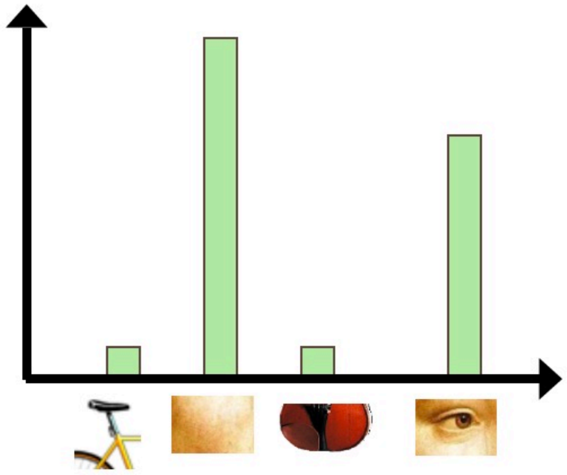
“Words” from cars

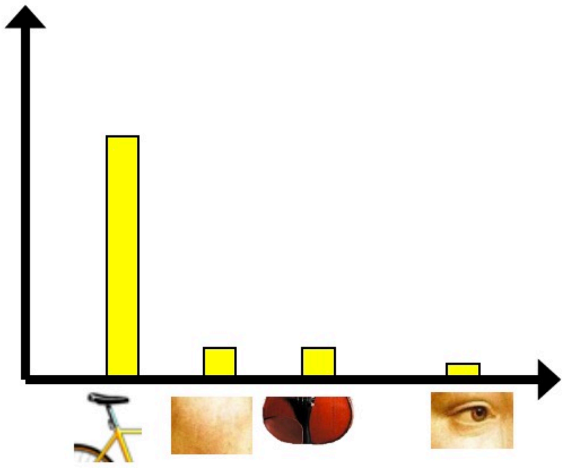


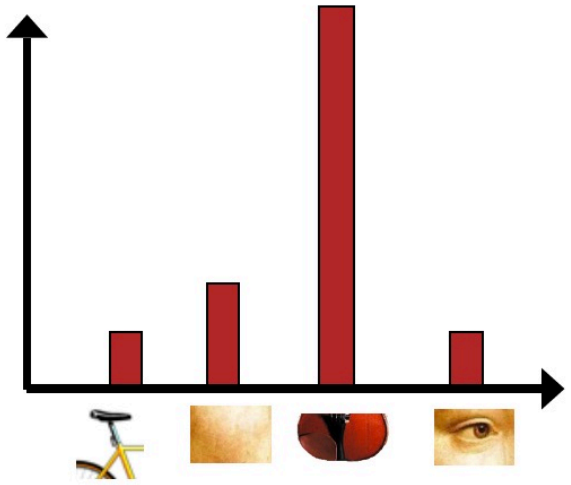


Histogram of Visual Words

- Count how many times each “word” appears in an image -> Histogram of words
- Can you tell the image types from the following histograms?







From http://www.cs.cmu.edu/~16385/lectures/Lecture12.pdf

How to Classify Images?

- Use the histogram of words as the descriptor for an image
- Train a classifier (e.g. Support Vector Machine) from training images

Pop Quiz

- What are the major steps to use BoW to classify images?
  - Training stage?
  - How to classify a test image?

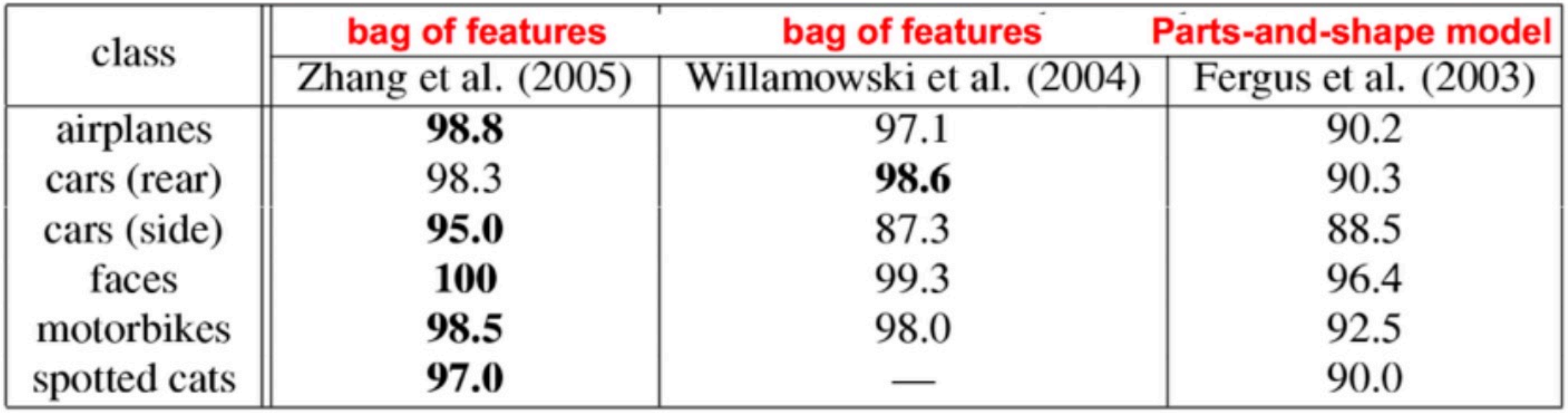
Pop Quiz

- What are the major steps to use BoW to classify images?
- Training stage
  - For each training image, detect feature points and generate the feature descriptors;
  - Using K-means to cluster all the descriptors from all training images to a dictionary of words;
  - Express each training image by a histogram of words (BoW);
  - Train a classifier (SVM or NN) using the BoW representations.
- Given a test image
  - Generate Gaussian and Laplacian scale images
  - Detect feature points
  - Generate descriptors at detected feature points
  - Quantize each descriptor to the nearest word in the dictionary
  - Generate the histogram of words for the image
  - Apply the classifier

Power of BoW!

Now beaten by Deep Networks :(

CalTech6 dataset



class	bag of features Zhang et al. (2005)	bag of features Willamowski et al. (2004)	Parts-and-shape model Fergus et al. (2003)
airplanes	**98.8**	97.1	90.2
cars (rear)	98.3	**98.6**	90.3
cars (side)	**95.0**	87.3	88.5
faces	**100**	99.3	96.4
motorbikes	**98.5**	98.0	92.5
spotted cats	**97.0**	—	90.0

Works pretty well for image-level classification

From http://www.cs.cmu.edu/~16385/lectures/Lecture12.pdf

Summary

- Criteria to select features and describe features
  - Invariance to view angle change (shift, rotation and scale, more generally affine transformation) and contrast change.
  - Robust to clutter and occlusion
    - By using local features
- Feature detectors
  - Locating corners where there are changes in multiple directions
    - Edges are not good features (not unique)
    - Harris corner detector
  - Can be observed in different scales (stable cross scales)
    - Generating Gaussian and Laplacian Scale space
    - SIFT detector: Detecting local extrema in Laplacian scale space
    - Harris-Laplacian: Harris detector in each Gaussian scale, detect characteristic scale of each feature

Summary (Cnt'd)

- Feature Descriptors
  - Achieving invariance to contrast and position shift by using histogram of orientation of gradient (HoG)
  - Achieving rotation invariance by shifting the dominant orientation in the HoG to the same reference orientation
  - Achieve scale invariance through detecting the characteristic scales
  - Most popular: SIFT, SURF
- Use of feature detection and descriptors
  - Image classification using Bag of Words
  - Feature matching for image registration/ stitching (next lecture)

What you should know?

- Be able to answer all the pop quizzes!

Reading Assignment

- [Szeliski2021] Richard Szeliski, Computer Vision: Algorithms and Applications. 2021. Sec. 7.1.1, 7.1.2, 6.2.1.
- **Optional:**
- Harris, C. and Stephens, M. J. (1988). A combined corner and edge detector. In Alvey Vision Conference , pp. 147–152.
- Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision , 60(2):91–110.
- Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., and Van Gool, L. J. (2005). A comparison of affine region detectors. International Journal of Computer Vision , 65(1-2):43–72.
  - Code for different detectors: http://www.robots.ox.ac.uk/vgg/research/affine/
- Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence , 27(10):1615–1630.
- Bay, H., Tuytelaars, T., and Van Gool, L. (2006). SURF: Speeded up robust features. In Ninth European Conference on Computer Vision (ECCV 2006) , pp. 404–417.
- An easy to follow tutorial: http://aishack.in/tutorials/sift-scale-invariant-feature-transform-introduction/

Software Tools

- VLFeat, an open and portable library of computer vision algorithms, http://vlfeat.org/.
  - Include both SIFT detector and descriptor
  - Written in C, with interface for MATLAB
  - Matlab : http://www.vlfeat.org/install-matlab.html (install tutorial)
- SiftGPU: A GPU Implementation of Scale Invariant Feature Transform (SIFT), http:
  - //www.cs.unc.edu/ccwu/siftgpu/ (Wu 2010).
- SURF: Speeded Up Robust Features, http://www.vision.ee.ethz.ch/surf/ (Bay, Tuytelaars, and Van Gool 2006).
  - Python : in opencv package. Simply call by cv2.SURF
- Harris corner detector :
  - Python : in opencv package. Simply call by cv2.cornerHarris
  - MATLAB: corner( )

Software Tools

- SIFT tools in python through opencv
  - detector = cv2.FeatureDetector_create("SIFT")
  - descriptor = cv2.DescriptorExtractor_create("SIFT")
  - skp = detector.detect(img)
  - skp, sd=descriptor.compute(img,skp)
  - https://jayrambhia.wordpress.com/2013/01/18/sift-keypoint-matching-using-python-opencv/

Review Questions

1. What are the desired properties of feature detectors?
2. What are the desired properties of feature descriptors?
3. Are points with high gradient magnitudes always good as a feature point?
4. What are the steps in detecting Harris feature points at a fixed scale?
5. What are the steps in generating a Laplacian of Gaussian scale space?
6. What are the steps in detecting Harris feature points at characteristic scales?
7. What are the steps in detecting SIFT feature points?
8. What are the steps in forming the Histogram of Orientation of Gradients (HoG) of a small patch?
9. What are the steps in forming the SIFT descriptor of a feature point, given its scale and orientation?
10. Show that difference of two Gaussian filtered images with scales $k\sigma$ and σ is proportional to an image convolved with a Laplacian of Gaussian filter at the scale of σ. Note that this is the basis for Lowe to propose to generate the scale space of normalized Laplacian of Gaussian filtered images using differences of Gaussian filtered images.

Written Homework

1. Consider the following images. For the center patch of each image (with size 5x5, colored in red), determine the moment matrix A and find its eigenvalues and eigenvectors and finally the Harris cornerness value. Furthermore, show that the Harris value computed using the eigenvalues are the same as that using the determinant and trace of A. Which patch has a higher Harris value? Is that as expected? For simplicity, use central difference to compute the horizontal and vertical gradient (i.e, filter [-1, 0, 1].) Show the gradient images as intermediate results. You can ignore the Gaussian weighting step when computing the A matrix.

$$\begin{array}{ccccccccc}
0&0&0&0&1&1&1&1&1\\
0&0&0&0&1&1&1&1&1\\
0&0&0&0&1&1&1&1&1\\
0&0&0&0&1&1&1&1&1\\
0&0&0&0&1&1&1&1&1\\
0&0&0&0&1&1&1&1&1\\
0&0&0&0&1&1&1&1&1\\
0&0&0&0&1&1&1&1&1\\
0&0&0&0&1&1&1&1&1
\end{array}
\qquad
\begin{array}{ccccccccc}
0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&1&1&1&1\\
0&0&0&0&1&1&1&1&1\\
0&0&0&0&1&1&1&1&1\\
0&0&0&0&1&1&1&1&1\\
0&0&0&0&1&1&1&1&1
\end{array}
\qquad
\begin{array}{ccccccccc}
0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&0\\
0&0&0&1&1&1&0&0&0\\
0&0&1&1&1&1&1&0&0\\
0&1&1&1&1&1&1&1&0\\
1&1&1&1&1&1&1&1&1
\end{array}$$

2. Compute the HoG for the center patch in each of the above images and determine the dominant orientations. Are your results as expected? Use 8 orientation bins for the HoG. You can ignore the Gaussian weighting.

For the above two problems you can do either manually or using MATLAB/Python or a combination.