Local Feature Detection

Faisal Qureshi
faisal.qureshi@uoit.ca
Interest Points

Slides from Derek Hoiem, Svetlana Lazebnik, Antonio Torralba, Steve Seitz, David Forsyth, David Lowe, Fei-Fei Li, and James Hays.
Motivation for Feature Extraction

Panorama Stitching
Requires that we find “corresponding locations” in two images to compute homography
How to find “corresponding locations” automatically?
How to find “corresponding locations” automatically?

Detection
Identify interest points
How to find “corresponding locations” automatically?

Detection
Identify interest points

Description
Extract features around interest points

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

\[ \mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}] \]
How to find “corresponding locations” automatically?

Detection
Identify interest points

Description
Extract features around interest points

Matching
Match features in two images
Characteristics of a Good Feature

- Repeatability — feature is invariant to geometric, lighting, etc. changes
- Saliency or distinctiveness
- Compactness — efficiency, many fewer features than the number of pixels in the image
- Locality — robustness to clutter and occlusion, a feature should only occupy a small area of an image
Characteristics of a Good Feature

- Repeatability — feature is invariant to geometric, lighting, etc. changes
- Saliency or distinctiveness
- Compactness — efficiency, many fewer features than the number of pixels in the image
- Locality — robustness to clutter and occlusion, a feature should only occupy a small area of an image
Repeatability

- We need to find at least some of the same points in two images to any chance of finding true matches.
Repeatability

- We need to find at least some of the same points in two images to any chance of finding true matches.

No chance of finding a true match
Repeatability

- We need to find at least some of the same points in two images to any chance of finding true matches.

Detection process run independently on two images should return at least some of the corresponding locations.
Characteristics of a Good Feature

- **Repeatability** — feature is invariant to geometric, lighting, etc. changes

- **Saliency or distinctiveness**

- **Compactness** — efficiency, many fewer features than the number of pixels in the image

- **Locality** — robustness to clutter and occlusion, a feature should only occupy a small area of an image
Distinctiveness

- We want to reliably determine which location in one image goes with which location in the second image
Distinctiveness

• We want to reliably determine which location in one image goes with which location in the second image.
Distinctiveness

- We want to reliably determine which location in one image goes with which location in the second image.
Distinctiveness

- We want to reliably determine which location in one image goes with which location in the second image.
Distinctiveness

- We want to reliably determine which location in one image goes with which location in the second image.
Distinctiveness

- We want to reliably determine which location in one image goes with which location in the second image.
Distinctiveness

- We want to reliably determine which location in one image goes with which location in the second image.
Distinctiveness

- We want to reliably determine which location in one image goes with which location in the second image.

Invariance to geometric and photometric differences between the two images.
Feature Points’ Uses

- Image alignment
- 3D reconstruction
- Tracking
- Object recognition
- Image retrieval
- Navigation and mapping
Local Features

• Part 1: detect interest points

• Part 2: compute descriptors that encode the area surrounding an interest point

• Part 3: use descriptors for finding matches (identifying corresponding locations, for example)
Local Features

• Part 1: detect interest points

• Part 2: compute descriptors that encode the area surrounding an interest point

• Part 3: use descriptors for finding matches (identifying corresponding locations, for example)
Many Existing Detectors Available

Hessian & Harris
Laplacian, DoG
Harris-/Hessian-Laplace
Harris-/Hessian-Affine
EBR and IBR
MSER
Salient Regions
Others...

[Beaudet ‘78], [Harris ‘88]
[Lindeberg ‘98], [Lowe 1999]
[Mikolajczyk & Schmid ‘01]
[Mikolajczyk & Schmid ‘04]
[Tuytelaars & Van Gool ‘04]
[Matas ‘02]
[Kadir & Brady ‘01]
Interest Point Detection

Available choices

- Hessian & Harris [Beaudet ‘78], [Harris ‘88]
- Laplacian, DoG [Lindeberg ‘98], [Lowe 1999]
- Harris-/Hessian-Laplace [Mikolajczyk & Schmid ‘01]
- Harris-/Hessian-Affine [Mikolajczyk & Schmid ‘04]
- EBR and IBR [Tuytelaars & Van Gool ‘04]
- MSER [Matas ‘02]
- Salient Regions [Kadir & Brady ‘01]
- and many others

From K. Grauman, B. Leibe
How to find an “interest point?”
How to find an “interest point?”
What is better, A or B?
How to find an “interest point?”
What is better, A or B?

Corners!
Corner Detection

• We should easily recognize the point looking through a small window

A
Flat area, difficult to localize

B
Corner, easy localization
Corner Detection
Corner Detection
Corner Detection

“flat” region: no change in all directions

Source: A. Efros
Corner Detection

“flat” region: no change in all directions

“edge”: no change along the edge direction

Source: A. Efros
Corner Detection

“flat” region: no change in all directions

“edge”: no change along the edge direction

“corner”: significant change in all directions

Source: A. Efros
Corner Detection

“flat” region: no change in all directions

“edge”: no change along the edge direction

“corner”: significant change in all directions

Shifting a window in either direction should give a large change in intensity

Source: A. Efros
Harris Corner Detection

- Corners are repeatable and distinctive
- Image gradients change in both directions around corners
Corner Detection: Mathematics

Change in appearance of window $w(x,y)$ for the shift $[u,v]$

$$E(u, v) = \sum_{x,y} w(x, y) [I(x + u, y + v) - I(x, y)]^2$$
Corner Detection: Mathematics

Change in appearance of window \( w(x,y) \) for the shift \([u,v]\)

\[
E(u, v) = \sum_{x,y} w(x, y) [I(x + u, y + v) - I(x, y)]^2
\]
Corner Detection: Mathematics

Change in appearance of window $w(x,y)$ for the shift $[u,v]$

$$E(u, v) = \sum_{x,y} w(x, y) \left[ I(x + u, y + v) - I(x, y) \right]^2$$
Corner Detection: Mathematics

Change in appearance of window \( w(x, y) \) for the shift \([u, v]\)

\[
E(u, v) = \sum_{x, y} w(x, y) [I(x + u, y + v) - I(x, y)]^2
\]
Corner Detection: Mathematics

Change in appearance of window $w(x,y)$ for the shift $[u,v]$

$$E(u, v) = \sum_{x,y} w(x, y) [I(x + u, y + v) - I(x, y)]^2$$
Corner Detection: Mathematics

Change in appearance of window \( w(x,y) \) for the shift \([u,v]\)

\[
E(u, v) = \sum_{x,y} w(x, y) [I(x + u, y + v) - I(x, y)]^2
\]

Source: R. Szeliski
Corner Detection: Mathematics

Change in appearance of window \( w(x,y) \) for the shift \([u,v]\)

\[
E(u,v) = \sum_{x,y} w(x,y) [I(x + u, y + v) - I(x, y)]^2
\]

Window function

Shifted intensity

**Window function** \( w(x,y) = \)

1 in window, 0 outside

or

Gaussian

Source: R. Szeliski
Corner Detection: Mathematics

• How does $E(u,v)$ behaves for small shifts $[u,v]$?

$$E(u, v) = \sum_{x,y} w(x, y) [I(x + u, y + v) - I(x, y)]^2$$

$E(u, v)$
Corner Detection: Mathematics

• How does $E(u,v)$ behaves for small shifts $[u,v]$?

$$E(u, v) = \sum_{x,y} w(x, y) [I(x + u, y + v) - I(x, y)]^2$$

• Use Taylor Series for local quadratic approximation of $E(u,v)$

$$E(u, v) = E(0, 0) + \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} E_u(0, 0) \\ E_v(0, 0) \end{bmatrix} + \frac{1}{2} \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} E_{uu}(0, 0) & E_{uv}(0, 0) \\ E_{vu}(0, 0) & E_{vv}(0, 0) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$
Corner Detection: Mathematics

\[ E(u, v) = E(0, 0) + \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} E_u(0, 0) \\ E_v(0, 0) \end{bmatrix} + \frac{1}{2} \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} E_{uu}(0, 0) & E_{uv}(0, 0) \\ E_{vu}(0, 0) & E_{vv}(0, 0) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \]

\[ E_u(u, v) = \sum_{x,y} 2w(x, y)[I(x + u, y + v) - I(x, y)]I_x(x + u, y + v) \]

\[ E_{uu}(u, v) = \sum_{x,y} 2w(x, y)I_x(x + u, y + v)I_x(x + u, y + v) \]
\[ + \sum_{x,y} 2w(x, y)[I(x + u, y + v) - I(x, y)]I_{xx}(x + u, y + v) \]

\[ E_{uv}(u, v) = \sum_{x,y} 2w(x, y)I_y(x + u, y + v)I_x(x + u, y + v) \]
\[ + \sum_{x,y} 2w(x, y)[I(x + u, y + v) - I(x, y)]I_{xy}(x + u, y + v) \]
Corner Detection: Mathematics

\[ E(u, v) = E(0, 0) + \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} E_u(0, 0) \\ E_v(0, 0) \end{bmatrix} + \frac{1}{2} \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} E_{uu}(0, 0) & E_{uv}(0, 0) \\ E_{vu}(0, 0) & E_{vv}(0, 0) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \]

\[ E(0, 0) = 0 \]
\[ E_u(0, 0) = 0 \]
\[ E_v(0, 0) = 0 \]

\[ E_{uu}(0, 0) = \sum_{x,y} 2w(x, y)I_x(x, y)I_x(x, y) \]

\[ E_{vv}(0, 0) = \sum_{x,y} 2w(x, y)I_y(x, y)I_y(x, y) \]

\[ E_{uv}(0, 0) = \sum_{x,y} 2w(x, y)I_x(x, y)I_y(x, y) \]
Corner Detection: Mathematics

• The quadratic approximation simplifies to

\[ E(u, v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} \]

• \( M \) is the second moment matrix

\[ M = \sum_{x,y} w(x, y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \]
Corners as Distinctive Interest Points

Image derivatives averaged in neighbourhood of a point
Corners as Distinctive Interest Points

Image derivatives averaged in neighbourhood of a point

\[ I_x \Leftrightarrow \frac{\partial I}{\partial x} \quad I_y \Leftrightarrow \frac{\partial I}{\partial y} \quad I_x I_y \Leftrightarrow \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \]
Interpreting the second moment matrix

The surface $E(u,v)$ is locally approximated by a quadratic form. Let’s try to understand its shape.

$$E(u,v) \approx [u \ v] \ M \ [u \ v]$$

$$M = \sum_{x,y} w(x, y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$
Interpreting the second moment matrix

First, consider the axis-aligned case (gradients are either horizontal or vertical)

\[
M = \sum_{x, y} w(x, y) \begin{bmatrix}
I_x^2 & I_x I_y \\
I_x I_y & I_y^2 \\
\end{bmatrix} = \begin{bmatrix}
\lambda_1 & 0 \\
0 & \lambda_2 \\
\end{bmatrix}
\]
Interpreting the second moment matrix

First, consider the axis-aligned case (gradients are either horizontal or vertical)

\[ M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \]

If either \( \lambda \) is close to 0, then this is not a corner, so look for locations where both are large.
Interpreting the second moment matrix

\[
\begin{bmatrix}
u & v
\end{bmatrix} \ M \ \begin{bmatrix}
u \\
v
\end{bmatrix} = \text{const}
\]

Consider a horizontal “slice” of $E(u, v)$:
Interpreting the second moment matrix

This is the equation of an ellipse: \[ [u \ v] \ M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const} \]

Consider a horizontal “slice” of \(E(u, v)\):
Interpreting the second moment matrix

This is the equation of an ellipse.  \[ \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const} \]

Diagonalization of M: \[ M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R \]

The axis lengths of the ellipse are determined by the eigenvalues and the orientation is determined by R.

direction of the fastest change

direction of the slowest change
Visualization of second moment matrices
Visualization of second moment matrices
Interpreting the eigenvalues

Classification of image points using eigenvalues of $M$:

- **“Corner”**: $\lambda_1$ and $\lambda_2$ are large, $\lambda_1 \sim \lambda_2$; $E$ increases in all directions
- **“Edge”**: $\lambda_2 \gg \lambda_1$
- **“Flat” region**: $\lambda_1$ and $\lambda_2$ are small; $E$ is almost constant in all directions

$\lambda_1$ and $\lambda_2$
Interpreting the eigenvalues

Classification of image points using eigenvalues of $M$:

- **Corner**: $\lambda_1$ and $\lambda_2$ are large, $\lambda_1 \sim \lambda_2$; $E$ increases in all directions.
- **Edge**: $\lambda_1 >> \lambda_2$; $E$ almost constant in all directions.
- **Flat region**: $\lambda_2 >> \lambda_1$; $E$ is almost constant in all directions.

The criterion is:

$$R = \det(M) - \alpha \text{trace}(M)^2 = \lambda_1 \lambda_2 - \alpha (\lambda_1 + \lambda_2)^2$$

$\alpha$: constant (0.04 to 0.06)
Harris corner detector

1) Compute $M$ matrix for each image window to get their *cornerness* scores.

2) Find points whose surrounding window gave large corner response ($f >$ threshold)

3) Take the points of local maxima, i.e., perform non-maximum suppression

Harris Detector: Steps
Harris Detector: Steps

Compute corner response $R$
Harris Detector: Steps

Find points with large corner response: $R > \text{threshold}$
Harris Detector: Steps

Take only the points of local maxima of $R$
Harris Detector: Steps
Invariance and covariance

• We want corner locations to be *invariant* to photometric transformations and *covariant* to geometric transformations

• Invariance: *image is transformed and corner locations do not change*

• Covariance: *if we have two transformed versions of the same image, features should be detected in corresponding locations*
Invariance and covariance
Affine intensity change

$I \rightarrow aI + b$
Affine intensity change

\[ I \rightarrow aI + b \]

Only derivatives are used => invariance to intensity shift \( I \rightarrow I + b \)
Affine intensity change

$I \rightarrow aI + b$

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

Intensity scaling: $I \rightarrow aI$

threshold

$x$ (image coordinate)

$R$

$x$ (image coordinate)
Affine intensity change

$I \rightarrow aI + b$

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

Intensity scaling: $I \rightarrow aI$

Partially invariant to affine intensity change

$R$

$x$ (image coordinate)

$R$

$x$ (image coordinate)

*Partially invariant* to affine intensity change
Image translation
Image translation

- Derivatives and window function are shift-invariant
Image translation

- Derivatives and window function are shift-invariant

Corner location is covariant w.r.t. translation
Image rotation
Image rotation
Image rotation

Second moment ellipse rotates but its shape (i.e. eigenvalues) remains the same
Second moment ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner location is covariant w.r.t. rotation
Scaling

Corner
Scaling

Corner
All points will be classified as edges.
All points will be classified as edges

Corner location is not covariant to scaling!
Feature extraction: Corners
Local Features

• Part 1: detect interest points

• Part 2: compute descriptors that encode the area surrounding an interest point

• Part 3: use descriptors for finding matches (identifying corresponding locations, for example)
Interest Points

Slides from Derek Hoiem, Svetlana Lazebnik, Antonio Torralba, Steve Seitz, David Forsyth, David Lowe, Fei-Fei Li, and James Hays.
Local Invariant Features

• Detection of interest points
  • (Harris corner detection)
  • Scale invariant blob detection: LoG

• Description of local patches
  • SIFT pipeline for invariant local features
Recall: Corners as Distinctive Interest Points

Since $M$ is symmetric, we have

$$M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T$$
Recall: Corners as Distinctive Interest Points

Since $M$ is symmetric, we have

$$M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T$$

$$Mx_i = \lambda_i x_i$$
Recall: Corners as Distinctive Interest Points

Since $M$ is symmetric, we have

\[
M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T
\]

\[
Mx_i = \lambda_i x_i
\]
Recall: Corners as Distinctive Interest Points

Since $M$ is symmetric, we have

$$
M = X \begin{bmatrix}
\lambda_1 & 0 \\
0 & \lambda_2
\end{bmatrix} X^T
$$

$$
Mx_i = \lambda_i x_i
$$

The *eigenvalues* of $M$ reveal the amount of intensity change in the two principal orthogonal gradient directions in the window.
Recall: Corners as Distinctive Interest Points

“edge”:

“corner”:

“flat” region
Recall: Corners as Distinctive Interest Points

“edge”: 
\[ \lambda_1 >> \lambda_2 \]
\[ \lambda_2 >> \lambda_1 \]

“corner”:

“flat” region
Recall: Corners as Distinctive Interest Points

“edge”: $\lambda_1 \gg \lambda_2$
$\lambda_2 \gg \lambda_1$

“corner”: $\lambda_1$ and $\lambda_2$ are large,
$\lambda_1 \sim \lambda_2$;

“flat” region
Recall: Corners as Distinctive Interest Points

“edge”:
\[ \lambda_1 >> \lambda_2 \]
\[ \lambda_2 >> \lambda_1 \]

“corner”:
\[ \lambda_1 \text{ and } \lambda_2 \text{ are large,} \]
\[ \lambda_1 \sim \lambda_2; \]

“flat” region
\[ \lambda_1 \text{ and } \lambda_2 \text{ are small;} \]
Recall: Corners as Distinctive Interest Points

“edge”:
\[ \lambda_1 \gg \lambda_2 \]
\[ \lambda_2 \gg \lambda_1 \]

“corner”:
\[ \lambda_1 \text{ and } \lambda_2 \text{ are large, } \lambda_1 \sim \lambda_2; \]

“flat” region
\[ \lambda_1 \text{ and } \lambda_2 \text{ are small; } \]

One way to score the cornerness:
\[ f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2} \]
Harris Detector [Harris88]

• Second moment matrix (autocorrelation matrix)

\[
\mu(\sigma_I, \sigma_D) = g(\sigma_I) \ast \begin{bmatrix}
I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\
I_x I_y(\sigma_D) & I_y^2(\sigma_D)
\end{bmatrix}
\]
Harris Detector [Harris88]

- Second moment matrix (autocorrelation matrix)
  \[
  \mu(\sigma_I, \sigma_D) = g(\sigma_I) \begin{bmatrix}
  I_x^2(\sigma_D) & I_xI_y(\sigma_D) \\
  I_xI_y(\sigma_D) & I_y^2(\sigma_D)
  \end{bmatrix}
  \]

1. Image derivatives
Harris Detector [Harris88]

- Second moment matrix (autocorrelation matrix)

\[
\mu(\sigma_I, \sigma_D) = g(\sigma_I) \ast \begin{bmatrix}
I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\
I_x I_y(\sigma_D) & I_y^2(\sigma_D)
\end{bmatrix}
\]

1. Image derivatives

2. Square of derivatives
Harris Detector [Harris88]

• Second moment matrix (autocorrelation matrix)

\[
\mu(\sigma_I, \sigma_D) = g(\sigma_I) \ast \begin{bmatrix}
I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\
I_x I_y(\sigma_D) & I_y^2(\sigma_D)
\end{bmatrix}
\]

1. Image derivatives

2. Square of derivatives

3. Gaussian filter \( g(\sigma_I) \)
Harris Detector [Harris88]

• Second moment matrix
  (autocorrelation matrix)
  \[
  \mu(\sigma_I, \sigma_D) = g(\sigma_I) \ast \begin{bmatrix}
  I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\
  I_x I_y(\sigma_D) & I_y^2(\sigma_D)
  \end{bmatrix}
  \]

  \[\det M = \lambda_1 \lambda_2\]
  \[\text{trace } M = \lambda_1 + \lambda_2\]

  1. Image derivatives

  2. Square of derivatives

  3. Gaussian filter \(g(\sigma_I)\)

  4. Cornerness function – both eigenvalues are strong
  \[
  har = \det[\mu(\sigma_I, \sigma_D)] - \alpha[\text{trace}(\mu(\sigma_I, \sigma_D))^2] =
  g(I_x^2)g(I_y^2) - [g(I_x I_y)]^2 - \alpha[g(I_x^2) + g(I_y^2)]^2
  \]

  5. Non-maxima suppression
Properties of the Harris corner detector

Rotation invariant?

Scale invariant?
Properties of the Harris corner detector

Rotation invariant?  Yes

Scale invariant?
Properties of the Harris corner detector

Rotation invariant?   Yes

Scale invariant?
Properties of the Harris corner detector

Rotation invariant?  Yes

Scale invariant?  No
Properties of the Harris corner detector

Rotation invariant? Yes

Scale invariant? No

All points will be classified as edges
Properties of the Harris corner detector

Rotation invariant? Yes

Scale invariant? No

All points will be classified as edges

Corner!
Scale invariant interest points

How can we independently select interest points in each image, such that the detections are repeatable across different scales?
Automatic scale selection
Automatic scale selection

Intuition:
• Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

**Intuition:**
• Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

**Intuition:**
- Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

**Intuition:**
- Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

Intuition:
• Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

**Intuition:**
- Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

Intuition:
• Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

Intuition:
• Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

**Intuition:**
- Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

Intuition:
• Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

Intuition:
- Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

Intuition:
• Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

Intuition:
• Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

Intuition:
• Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

**Intuition:**
- Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

**Intuition:**
- Find scale that gives local maxima of some function $f$ in both position and scale.
Automatic scale selection

**Intuition:**
- Find scale that gives local maxima of some function $f$ in both position and scale.

![Image 1](image1.png)

$\text{region size}$

$S_1$

![Image 2](image2.png)

$\text{region size}$

$S_2$
Recall: Edge detection

$f$

$\frac{d}{dx} g$

$f * \frac{d}{dx} g$

Edge = maximum of derivative

Source: S. Seitz
Recall: Edge detection

Edge = zero crossing of second derivative

**Source:** S. Seitz
From edges to blobs

- Edge = ripple
- Blob = superposition of two ripples
From edges to blobs

- Edge = ripple
- Blob = superposition of two ripples

Slide credit: Lana Lazebnik
From edges to blobs

- Edge = ripple
- Blob = superposition of two ripples

**Spatial selection**: the magnitude of the Laplacian response will achieve a maximum at the center of the blob, provided the scale of the Laplacian is “matched” to the scale of the blob.
Blob detection in 2D

Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D

\[ \nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \]
Blob detection in 2D: scale selection

Laplacian-of-Gaussian = “blob” detector

\[ \nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \]
Blob detection in 2D: scale selection

Laplacian-of-Gaussian = “blob” detector

$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$
Blob detection in 2D: scale selection

Laplacian-of-Gaussian = “blob” detector

$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$
Blob detection in 2D: scale selection

Laplacian-of-Gaussian = “blob” detector

\[ \nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \]
Blob detection in 2D

We define the *characteristic scale* as the scale that produces peak of Laplacian response.
Example

Original image at $\frac{3}{4}$ the size
Original image at $\frac{3}{4}$ the size
Kristen Grauman
Scale invariant interest points

Interest points are local maxima in both position and scale.

\[ L_{xx}(\sigma) + L_{yy}(\sigma) \Rightarrow \text{List of } (x, y, \sigma) \]

Squared filter response maps
Scale-space blob detector: Example

Image credit: Lana Lazebnik
Technical detail

We can approximate the Laplacian with a difference of Gaussians; more efficient to implement.

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

\[ DoG = G(x, y, k\sigma) - G(x, y, \sigma) \]  
(Difference of Gaussians)
Technical detail

We can approximate the Laplacian with a difference of Gaussians; more efficient to implement.

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

(Laplacian)

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

(Difference of Gaussians)
Maximally Stable Extremal Regions [Matas ‘02]
• Based on Watershed segmentation algorithm
• Select regions that stay stable over a large parameter range
Maximally Stable Extremal Regions [Matas ‘02]

• Based on Watershed segmentation algorithm
• Select regions that stay stable over a large parameter range
Example Results: MSER
Comparison

LoG

MSER

Harris
# Comparison of Keypoint Detectors

<table>
<thead>
<tr>
<th>Feature Detector</th>
<th>Corner</th>
<th>Blob</th>
<th>Region</th>
<th>Rotation invariant</th>
<th>Scale invariant</th>
<th>Affine invariant</th>
<th>Repeatability</th>
<th>Localization accuracy</th>
<th>Robustness</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Harris</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+++</td>
<td>+++</td>
<td>+++</td>
<td>+</td>
</tr>
<tr>
<td>Hessian</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>++</td>
<td>++</td>
<td>✓</td>
<td>+</td>
</tr>
<tr>
<td>SUSAN</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>Harris-Laplace</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+++</td>
<td>+++</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>Hessian-Laplace</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+++</td>
<td>+++</td>
<td>+++</td>
<td>+</td>
</tr>
<tr>
<td>DoG</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>SURF</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>Harris-Affine</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+++</td>
<td>+++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>Hessian-Affine</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+++</td>
<td>+++</td>
<td>+++</td>
<td>+</td>
</tr>
<tr>
<td>Salient Regions</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>++</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>Edge-based</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+++</td>
<td>+++</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>MSER</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+++</td>
<td>+++</td>
<td>++</td>
<td>+++</td>
</tr>
<tr>
<td>Intensity-based</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>Superpixels</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

Tuytelaars Mikolajczyk 2008
Choosing a detector

• What do you want it for?
  • Precise localization in x-y: Harris
  • Good localization in scale: Difference of Gaussian
  • Flexible region shape: MSER

• Best choice often application dependent
Choosing a detector

• What do you want it for?
  • Precise localization in x-y: Harris
  • Good localization in scale: Difference of Gaussian
  • Flexible region shape: MSER

• Best choice often application dependent
  • Harris-/Hessian-Laplace/DoG work well for many natural categories
  • MSER works well for buildings and printed things

• Why choose?
Choosing a detector

• What do you want it for?
  • Precise localization in x-y: Harris
  • Good localization in scale: Difference of Gaussian
  • Flexible region shape: MSER

• Best choice often application dependent
  • Harris-/Hessian-Laplace/DoG work well for many natural categories
  • MSER works well for buildings and printed things

• Why choose?
  • Get more points with more detectors

• There have been extensive evaluations/comparisons
Choosing a detector

• What do you want it for?
  • Precise localization in x-y: Harris
  • Good localization in scale: Difference of Gaussian
  • Flexible region shape: MSER

• Best choice often application dependent
  • Harris-/Hessian-Laplace/DoG work well for many natural categories
  • MSER works well for buildings and printed things

• Why choose?
  • Get more points with more detectors

• There have been extensive evaluations/comparisons
  • [Mikolajczyk et al., IJCV’05, PAMI’05]
  • All detectors/descriptors shown here work well
• For most local feature detectors, executables are available online:
  – http://robots.ox.ac.uk/~vgg/research/affine
  – http://www.cs.ubc.ca/~lowe/keypoints/
  – http://www.vision.ee.ethz.ch/~surf
Descriptors
Local features: main components

1) Detection: Identify the interest points

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

\[ x_1 = [x_1^{(1)}, \ldots, x_d^{(1)}] \]

3) Matching: Determine correspondence between descriptors in two views

\[ x_2 = [x_1^{(2)}, \ldots, x_d^{(2)}] \]
Geometric transformations

e.g. scale, translation, rotation
Geometric transformations

- e.g. scale,
- translation,
- rotation

*Multiple View Geometry in computer vision* by Richard Hartley and Andrew Zisserman
Photometric transformations

Figure from T. Tuytelaars ECCV 2006 tutorial
Raw patches as local descriptors
Raw patches as local descriptors

The simplest way to describe the neighborhood around an interest point is to write down the list of intensities to form a feature vector.

But this is very sensitive to even small shifts, rotations.
Raw patches as local descriptors

The simplest way to describe the neighborhood around an interest point is to write down the list of intensities to form a feature vector.

But this is very sensitive to even small shifts, rotations.
Raw patches as local descriptors

The simplest way to describe the neighborhood around an interest point is to write down the list of intensities to form a feature vector.

But this is very sensitive to even small shifts, rotations.
Raw patches as local descriptors

The simplest way to describe the neighborhood around an interest point is to write down the list of intensities to form a feature vector.

But this is very sensitive to even small shifts, rotations.
Raw patches as local descriptors

The simplest way to describe the neighborhood around an interest point is to write down the list of intensities to form a feature vector.

But this is very sensitive to even small shifts, rotations.
SIFT descriptor [Lowe 2004]

• Use histograms to bin pixels within sub-patches according to their orientation.
SIFT descriptor [Lowe 2004]

- Use histograms to bin pixels within sub-patches according to their orientation.
SIFT descriptor [Lowe 2004]

- Use histograms to bin pixels within sub-patches according to their orientation.

![Diagram showing the process of using histograms to bin pixels within sub-patches according to their orientation.](image)
SIFT descriptor [Lowe 2004]

- Use histograms to bin pixels within sub-patches according to their orientation.
SIFT descriptor [Lowe 2004]

- Use histograms to bin pixels within sub-patches according to their orientation.

Why subpatches?
Why does SIFT have some illumination invariance?
Making descriptor rotation invariant

- Rotate patch according to its dominant gradient orientation
- This puts the patches into a canonical orientation.

Image from Matthew Brown
SIFT descriptor [Lowe 2004]

• 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
Example

NASA Mars Rover images
Example

NASA Mars Rover images with SIFT feature matches
Figure by Noah Snavely
SIFT properties

• Invariant to
  – Scale
  – Rotation

• Partially invariant to
  – Illumination changes
  – Camera viewpoint
  – Occlusion, clutter
Local features: main components

1) Detection: Identify the interest points

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

3) Matching: Determine correspondence between descriptors in two views
Matching local features
Matching local features

Image 1

Image 2
Matching local features
Matching local features

Image 1

Image 2

Kristen Grauman
Matching local features

To generate candidate matches, find patches that have the most similar appearance (e.g., lowest SSD)
Simplest approach: compare them all, take the closest (or closest k, or within a thresholded distance)
Ambiguous matches

Image 1

Image 2
Ambiguous matches

Image 1

Image 2

Kristen Grauman
Ambiguous matches

Image 1

Image 2

Kristen Grauman
Ambiguous matches

At what SSD value do we have a good match?
To add robustness to matching, can consider ratio: distance to best match / distance to second best match
If low, first match looks good.
If high, could be ambiguous match.
Matching SIFT Descriptors

- Nearest neighbor (Euclidean distance)
- Threshold ratio of nearest to $2^{nd}$ nearest descriptor
Recap: robust feature-based alignment

Source: L. Lazebnik
Recap: robust feature-based alignment

- Extract features

Source: L. Lazebnik
Recap: robust feature-based alignment

- Extract features
- Compute *putative matches*

Source: L. Lazebnik
Recap: robust feature-based alignment

- Extract features
- Compute *putative matches*
- Loop:
  - Hypothesize transformation $T$ (small group of putative matches that are related by $T$)

Source: L. Lazebnik
Recap: robust feature-based alignment

- Extract features
- Compute *putative matches*
- Loop:
  - Hypothesize transformation $T$ (small group of putative matches that are related by $T$)
  - Verify transformation (search for other matches consistent with $T$)

Source: L. Lazebnik
Recap: robust feature-based alignment

- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation $T$ (small group of putative matches that are related by $T$)
  - *Verify* transformation (search for other matches consistent with $T$)

Source: L. Lazebnik
Applications of local invariant features

• Wide baseline stereo
• Motion tracking
• Panoramas
• Mobile robot navigation
• 3D reconstruction
• Recognition
• …
Automatic mosaicing

http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html
Wide baseline stereo
Recognition of specific objects, scenes

Schmid and Mohr 1997

Sivic and Zisserman, 2003

Rothganger et al. 2003

Lowe 2002

Kristen Grauman
Summary

• Interest point detection
  – Harris corner detector
  – Laplacian of Gaussian, automatic scale selection

• Invariant descriptors
  – Rotation according to dominant gradient direction
  – Histograms for robustness to small shifts and translations (SIFT descriptor)
Local features
Specific recognition tasks
Scene categorization or classification

- outdoor/indoor
- city/forest/factory/etc.

Svetlana Lazebnik
Image annotation / tagging / attributes

- street
- people
- building
- mountain
- tourism
- cloudy
- brick
- ...

Svetlana Lazebnik
Object detection

• find pedestrians

Svetlana Lazebnik
Image parsing

sky
mountain
building
tree
banner
building
street lamp
market
people

Svetlana Lazebnik
Local features and bag of words models

• **Representation**
  – Gist descriptor
  – Image histograms
  – Sift-like features

• **Bag of Words models**
  – Encoding methods
Image Categorization

Training

Training Images

Training Labels

Derek Hoiem
Image Categorization

Training Images

Training

Image Features

Training Labels

Derek Hoiem
Image Categorization

Training Images

Training

Image Features

Classifier Training

Training Labels
Image Categorization

Training Images

Image Features

Classifier Training

Trained Classifier

Training Labels
Image Categorization

Training

Training Images → Image Features → Classifier Training → Trained Classifier

Test Image

Training Labels
Image Categorization

Training

Training Images

Training Labels

Classifier Training

Trained Classifier

Testing

Test Image

Image Features

Image Features
Image Categorization

Training

Training Images

Training Labels

Image Features

Classifier Training

Trained Classifier

Testing

Test Image

Image Features

Trained Classifier

Derek Hoiem
Image Categorization

Training
- Training Images
- Image Features
- Classifier Training
- Training Labels
- Trained Classifier

Testing
- Test Image
- Image Features
- Trained Classifier
- Prediction Outdoor

Derek Hoiem
Part 1: Image features

Training Images

Training

Image Features

Training Labels

Classifier Training

Trained Classifier

Derek Hoiem
Image Representations: Histograms

Global histogram

- Represent distribution of features
  - Color, texture, depth, ...
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
Computing histogram distance

\[
\text{histint}(h_i, h_j) = 1 - \sum_{m=1}^{K} \min(h_i(m), h_j(m))
\]

Histogram intersection (assuming normalized histograms)
Computing histogram distance

\[
\text{histint}(h_i, h_j) = 1 - \sum_{m=1}^{K} \min(h_i(m), h_j(m))
\]

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

James Hayes
Computing histogram distance

\[
\text{histint}(h_i, h_j) = 1 - \sum_{m=1}^{K} \min(h_i(m), h_j(m))
\]

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
Image Representations: Histograms

Clustering

Use the same cluster centers for all images

Images from Dave Kauchak
Histograms: Implementation issues

• Quantization
  – Grids: fast but applicable only with few dimensions
  – Clustering: slower but can quantize data in higher dimensions
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
Histograms: Implementation issues

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

- **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

Few Bins
Need less data
Coarser representation

Many Bins
Need more data
Finer representation
What kind of things do we compute histograms of?

- Color
  - L*a*b* color space
  - HSV color space

- Texture (filter banks or HOG over regions)
What kind of things do we compute histograms of?

• Histograms of oriented gradients

SIFT – Lowe IJCV 2004
SIFT vector formation

• Computed on rotated and scaled version of window according to computed orientation & scale
  – resample the window

• Based on gradients weighted by a Gaussian of variance half the window (for smooth falloff)
SIFT vector formation

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

showing only 2x2 here but is 4x4
Reduce effect of illumination

• 128-dim vector normalized to 1
• Threshold gradient magnitudes to avoid excessive influence of high gradients
  – after normalization, clamp gradients >0.2
  – renormalize
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

Example descriptor

Compute edges at four orientations
Extract a patch in each channel

Apply spatially varying blur and sub-sample

(Idealized signal)

Berg & Malik, CVPR 2001

K. Grauman, B. Leibe

James Hayes
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
Right features depend on what you want to know

- Shape: scene-scale, object-scale, detail-scale
- 2D form, shading, shadows, texture, linear perspective
Right features depend on what you want to know

- Shape: scene-scale, object-scale, detail-scale
  - 2D form, shading, shadows, texture, linear perspective

- Material properties: albedo, feel, hardness, ...
  - Color, texture
Right features depend on what you want to know

- Shape: scene-scale, object-scale, detail-scale
  - 2D form, shading, shadows, texture, linear perspective

- Material properties: albedo, feel, hardness, …
  - Color, texture

- Motion
  - Optical flow, tracked points
Right features depend on what you want to know

- **Shape**: scene-scale, object-scale, detail-scale
  - 2D form, shading, shadows, texture, linear perspective

- **Material properties**: albedo, feel, hardness, …
  - Color, texture

- **Motion**
  - Optical flow, tracked points

- **Distance**
  - Stereo, position, occlusion, scene shape
  - If known object: size, other objects
Things to remember about representation

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

• Think about the right features for the problem
  • Coverage
  • Concision
  • Directness
Bag-of-features models
Bag-of-features models
Origin 1: Texture recognition
Texture recognition

• Texture is characterized by the repetition of basic elements or *textons*
Origin 1: Texture recognition

• Texture is characterized by the repetition of basic elements or textons
• For stochastic textures, it is the identity of the textons, not their spatial arrangement, that matters
Origin 1: Texture recognition

- Texture is characterized by the repetition of basic elements or *textons*
- For stochastic textures, it is the identity of the textons, not their spatial arrangement, that matters

Origin 1: Texture recognition

Universal texton dictionary

histogram
Origin 1: Texture recognition

Origin 2: Bag-of-words models

Origin 2: Bag-of-words models

Origin 2: Bag-of-words models


US Presidential Speeches Tag Cloud
http://chir.ag/phernalia/preztags/

James Hayes
Origin 2: Bag-of-words models

- Orderless document representation: frequencies of words from a dictionary  
  Salton & McGill (1983)

US Presidential Speeches Tag Cloud  
http://chir.ag/phernalia/preztags/
Bag-of-features steps

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

- Regular grid or interest regions
1. Feature extraction

- Compute descriptor
- Normalize patch
- Detect patches

Slide credit: Josef Sivic
1. Feature extraction
2. Learning the visual vocabulary

Slide credit: Josef Sivic
2. Learning the visual vocabulary
2. Learning the visual vocabulary

Clustering
2. Learning the visual vocabulary

Visual vocabulary

Clustering
K-means clustering

\[ D(X, M) = \sum_{\text{cluster } k} \sum_{\text{point } i \text{ in cluster } k} (x_i - m_k)^2 \]
K-means clustering

• Want to minimize sum of squared Euclidean distances between points $x_i$ and their nearest cluster centers $m_k$

$$D(X, M) = \sum_{\text{cluster } k} \sum_{\text{point } i \text{ in cluster } k} (x_i - m_k)^2$$
K-means clustering

• Want to minimize sum of squared Euclidean distances between points $x_i$ and their nearest cluster centers $m_k$

\[
D(X, M) = \sum_{\text{cluster } k} \sum_{\text{point } i \text{ in cluster } k} (x_i - m_k)^2
\]

Algorithm:
K-means clustering

- Want to minimize sum of squared Euclidean distances between points $x_i$ and their nearest cluster centers $m_k$

\[
D(X, M) = \sum_{\text{cluster } k} \sum_{\text{point } i \text{ in cluster } k} (x_i - m_k)^2
\]

Algorithm:
- Randomly initialize K cluster centers
K-means clustering

- Want to minimize sum of squared Euclidean distances between points $x_i$ and their nearest cluster centers $m_k$

\[
D(X, M) = \sum_{\text{cluster } k} \sum_{\text{point } i \text{ in cluster } k} (x_i - m_k)^2
\]

Algorithm:
- Randomly initialize K cluster centers
- Iterate until convergence:
  - Assign each data point to the nearest center
  - Recompute each cluster center as the mean of all points assigned to it
Clustering and vector quantization
Clustering and vector quantization

- Clustering is a common method for learning a visual vocabulary or codebook
Clustering and vector quantization

- Clustering is a common method for learning a visual vocabulary or codebook
  - Unsupervised learning process
Clustering and vector quantization

• Clustering is a common method for learning a visual vocabulary or codebook
  • Unsupervised learning process
  • Each cluster center produced by k-means becomes a codevector
Clustering and vector quantization

• Clustering is a common method for learning a visual vocabulary or codebook
  • Unsupervised learning process
  • Each cluster center produced by k-means becomes a codevector
  • Codebook can be learned on separate training set
Clustering and vector quantization

- Clustering is a common method for learning a visual vocabulary or codebook
  - Unsupervised learning process
  - Each cluster center produced by k-means becomes a codevector
  - Codebook can be learned on separate training set
  - Provided the training set is sufficiently representative, the codebook will be "universal"
Clustering and vector quantization

- Clustering is a common method for learning a visual vocabulary or codebook
  - Unsupervised learning process
  - Each cluster center produced by k-means becomes a codevector
  - Codebook can be learned on separate training set
  - Provided the training set is sufficiently representative, the codebook will be “universal”

- The codebook is used for quantizing features
Clustering and vector quantization

• Clustering is a common method for learning a visual vocabulary or codebook
  • Unsupervised learning process
  • Each cluster center produced by k-means becomes a codevector
  • Codebook can be learned on separate training set
  • Provided the training set is sufficiently representative, the codebook will be “universal”

• The codebook is used for quantizing features
  • A vector quantizer takes a feature vector and maps it to the index of the nearest codevector in a codebook
Clustering and vector quantization

- Clustering is a common method for learning a visual vocabulary or codebook
  - Unsupervised learning process
  - Each cluster center produced by k-means becomes a codevector
  - Codebook can be learned on separate training set
  - Provided the training set is sufficiently representative, the codebook will be “universal”

- The codebook is used for quantizing features
  - A *vector quantizer* takes a feature vector and maps it to the index of the nearest codevector in a codebook
  - Codebook = visual vocabulary
Clustering and vector quantization

- Clustering is a common method for learning a visual vocabulary or codebook
  - Unsupervised learning process
  - Each cluster center produced by k-means becomes a codevector
  - Codebook can be learned on separate training set
  - Provided the training set is sufficiently representative, the codebook will be “universal”

- The codebook is used for quantizing features
  - A vector quantizer takes a feature vector and maps it to the index of the nearest codevector in a codebook
  - Codebook = visual vocabulary
  - Codevector = visual word
Example codebook

Source: B. Leibe
Another codebook

Appearance codebook

Source: B. Leibe
Another codebook

Appearance codebook

Source: B. Leibe
Visual vocabularies: Issues
Visual vocabularies: Issues

- How to choose vocabulary size?
Visual vocabularies: Issues

- How to choose vocabulary size?
  - Too small: visual words not representative of all patches
Visual vocabularies: Issues

• How to choose vocabulary size?
  • Too small: visual words not representative of all patches
  • Too large: quantization artifacts, overfitting
Visual vocabularies: Issues

• How to choose vocabulary size?
  • Too small: visual words not representative of all patches
  • Too large: quantization artifacts, overfitting

• Computational efficiency
Visual vocabularies: Issues

- How to choose vocabulary size?
  - Too small: visual words not representative of all patches
  - Too large: quantization artifacts, overfitting

- Computational efficiency
  - Vocabulary trees
    (Nister & Stewenius, 2006)
But what about layout?

All of these images have the same color histogram.
Spatial pyramid

Compute histogram in each spatial bin
Spatial pyramid

Compute histogram in each spatial bin
Spatial pyramid representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution

Lazebnik, Schmid & Ponce (CVPR 2006)
Spatial pyramid representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution

Lazebnik, Schmid & Ponce (CVPR 2006)
Spatial pyramid representation

• Extension of a bag of features
• Locally orderless representation at several levels of resolution

Lazebnik, Schmid & Ponce (CVPR 2006)
Spatial pyramid representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution

Lazebnik, Schmid & Ponce (CVPR 2006)
Spatial pyramid representation

• Extension of a bag of features
• Locally orderless representation at several levels of resolution

Lazebnik, Schmid & Ponce (CVPR 2006)
Spatial pyramid representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution

Lazebnik, Schmid & Ponce (CVPR 2006)
### Scene category dataset

#### Multi-class classification results
(100 training images per class)

<table>
<thead>
<tr>
<th>Level</th>
<th>Weak features (vocabulary size: 16)</th>
<th>Strong features (vocabulary size: 200)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Single-level</td>
<td>Pyramid</td>
</tr>
<tr>
<td>0 (1 x 1)</td>
<td>45.3 ±0.5</td>
<td></td>
</tr>
<tr>
<td>1 (2 x 2)</td>
<td>53.6 ±0.3</td>
<td>56.2 ±0.6</td>
</tr>
<tr>
<td>2 (4 x 4)</td>
<td>61.7 ±0.6</td>
<td>64.7 ±0.7</td>
</tr>
<tr>
<td>3 (8 x 8)</td>
<td>63.3 ±0.8</td>
<td><strong>66.8 ±0.6</strong></td>
</tr>
</tbody>
</table>
Caltech101 dataset

Caltech101 dataset


Multi-class classification results (30 training images per class)

<table>
<thead>
<tr>
<th>Level</th>
<th>Weak features (16)</th>
<th>Strong features (200)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Single-level</td>
<td>Pyramid</td>
</tr>
<tr>
<td>0</td>
<td>15.5 ± 0.9</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>31.4 ± 1.2</td>
<td>32.8 ± 1.3</td>
</tr>
<tr>
<td>2</td>
<td>47.2 ± 1.1</td>
<td>49.3 ± 1.4</td>
</tr>
<tr>
<td>3</td>
<td>52.2 ± 0.8</td>
<td>54.0 ± 1.1</td>
</tr>
</tbody>
</table>
Bags of features for action recognition

Space-time interest points

Summary

- Local features
- Bag of Word Models