Local invariant feature detection

Tues Feb 27
Kristen Grauman
UT Austin

Survey feedback

- Generally like
  - Assignments
  - Topics
  - Lecture engaging, like examples, interactive nature

- Lecture can be fast
  - Would like discussion section, more review
  - Careful about tangential questions
  - Questions are on slides but answers not written there too
  - Would like to videotape lectures for review later

- Content:
  - Programming (would like more) vs. math (difficult)
  - Grading: make sure fair partial credit
  - Book can be difficult to follow
  - Website:
    - Add direct link to current lecture (we have this)
    - Add TA emails (now added)
  - My office hours:
    - Schedule with me if you can’t make standard window

Review: Segmentation with texture

- Find “textons” by clustering vectors of filter bank outputs
- Describe texture in a window based on texton histogram

Segments as primitives for recognition

Multiple segmentations

B. Russell et al., “Using Multiple Segmentations to Discover Objects and their Extent in Image Collections,” CVPR 2006

Category-agnostic object “proposals”


Top-down segmentation

Top-down segmentation


Joint segmentation and recognition

Mask R-CNN, K. He et al., ICCV 2017

Video object segmentation

Goal: Extract all foreground objects
✓ even those unseen during training
✓ without manual intervention.

S. Jain et al., FusionSeg: Learning to combine motion and appearance for fully automatic segmentation of generic objects in videos, CVPR 2017

Interactive image and video segmentation

Results achieved with average of 2 user clicks

[Join & Grauman, HCOMP 2016] Click Carving
https://github.com/suyogduttjain/click_carving

Previously: Features and filters

Transforming and describing images; textures, colors, edges

Previously: Grouping & fitting

Clustering, segmentation, fitting; what parts belong together?
Now: Multiple views

Matching, invariant features, stereo vision, instance recognition

Important tool for multiple views: Local features

Multi-view matching relies on **local feature** correspondences.

How to detect **which local features** to match?

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

Local features: desired properties

- **Repeatability**
  - The same feature can be found in several images despite geometric and photometric transformations
- **Saliency**
  - Each feature has a distinctive description
- **Compactness and efficiency**
  - Many fewer features than image pixels
- **Locality**
  - A feature occupies a relatively small area of the image; robust to clutter and occlusion

Goal: interest operator repeatability

- We want to detect (at least some of) the same points in both images.

- Yet we have to be able to run the detection procedure **independently per image**.

No chance to find true matches!

Goal: descriptor distinctiveness

- We want to be able to reliably determine which point goes with which.

- Must provide some invariance to geometric and photometric differences between the two views.
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

Detecting corners
Compute “cornerness” response at every pixel.

Detecting local invariant features
- Detection of interest points
  - Harris corner detection
  - Scale invariant blob detection: LoG
- (Next time: description of local patches)
Corners as distinctive interest points
We should easily recognize the point by looking through a small window. Shifting a window in any direction should give a large change in intensity.

"flat" region: no change in all directions
"edge": no change along the edge direction
"corner": significant change in all directions

What does this matrix reveal?
First, consider an axis-aligned corner:

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

2 x 2 matrix of image derivatives (averaged in neighborhood of a point).

Notation:
\[ I_x \leftrightarrow \frac{\partial I}{\partial x}, \quad I_y \leftrightarrow \frac{\partial I}{\partial y}, \quad I_x I_y \leftrightarrow \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \]

What does this matrix reveal?
First, consider an axis-aligned corner:

\[
M = \sum \begin{bmatrix}
I_x^2 & I_x I_y \\
I_x I_y & I_y^2
\end{bmatrix} = \begin{bmatrix}
\lambda_1 & 0 \\
0 & \lambda_2
\end{bmatrix}
\]

This means dominant gradient directions align with x or y axis. Look for locations where both \(\lambda\)'s are large. If either \(\lambda\) is close to 0, then this is not corner-like.

What if we have a corner that is not aligned with the image axes?

The eigenvalues of \(M\) reveal the amount of intensity change in the two principal orthogonal gradient directions in the window.

Corner response function

\[
f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}
\]

Cornerness score (other variants possible)

"edge": \(\lambda_1 \gg \lambda_2\); \(\lambda_2 \gg \lambda_1\)
"corner": \(\lambda_1\) and \(\lambda_2\) are large, \(\lambda_1 - \lambda_2\)
"flat" region: \(\lambda_1, \lambda_2\) are small;
Harris corner detector

1) Compute $M$ matrix for each image window to get their corneress 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

Compute corner response $f$

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

Take only the points of local maxima of $f$
### Properties of the Harris corner detector

<table>
<thead>
<tr>
<th>Rotation invariant?</th>
<th>Yes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scale invariant?</td>
<td>No</td>
</tr>
</tbody>
</table>

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

### 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

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

![Region size](image1)

\[ s_1 \quad \text{region size} \]

\[ f \]

Image 1

![Region size](image2)

\[ s_2 \quad \text{region size} \]

\[ f \]

Image 2

**Function responses for increasing scale (scale signature)**
Automatic Scale Selection
- Function responses for increasing scale (scale signature)

What can be the “signature” function?
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

\[ \nabla^2 g = \frac{\partial^2 g}{\partial x} + \frac{\partial^2 g}{\partial y} \]

Blob detection in 2D

- We define the characteristic scale as the scale that produces peak of Laplacian response

Example

- Original image at ¾ the size

Slide credit: Kristen Grauman

Slide credit: Kristen Grauman

Slide credit: Kristen Grauman

Slide credit: Kristen Grauman
Scale-invariant interest points

Interest points are local maxima in both position and scale.

\[
L_x(x, y, \sigma) = L_y(x, y, \sigma) = 0
\]

\[
\text{List of (x, y, \sigma)}
\]

Scale-space blob detector: Example

Scale-space blob detector: Example

Technical detail

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

\[ L = \sigma^2 \left( G(x, y, \sigma) + G(x - k\sigma, y, \sigma) \right) \]  

(Laplacian)

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

(Difference of Gaussians)

Summary

- Desirable properties for local features for correspondence
- Basic matching pipeline
- Interest point detection
  - Harris corner detector
  - Laplacian of Gaussian, automatic scale selection

Local features: main components

1) Detection: Identify the interest points

**NEXT TIME**

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

3) Matching: Determine correspondence between descriptors in two views