CS4670/5670: Computer Vision
Kavita Bala

Lecture 13: Feature Descriptors and Matching
Announcements

• PA 2 out
• Artifact voting out: please vote

• Schedule will be updated shortly

• HW 1 out tonight
  – No slip days for HW 1
  – Due on Sunday Mar 13, answers released on Monday
Another type of feature

- The Laplacian of Gaussian (LoG)

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

(very similar to a Difference of Gaussians (DoG) – i.e. a Gaussian minus a slightly smaller Gaussian)
Scale Invariant Detection

- Functions for determining scale \( f = \text{Kernel} \ast \text{Image} \)

Kernels:

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

(Laplacian)

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

(Difference of Gaussians)

where Gaussian

\[
G(x, y, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2 + y^2}{2\sigma^2}}
\]

Note: both kernels are invariant to scale and rotation
Scale Invariant Detectors

- **Harris-Laplacian**\(^1\)
  *Find local maximum of:*
  - Harris corner detector in space (image coordinates)
  - Laplacian in scale

- **SIFT (Lowe)**\(^2\)
  *Find local maximum of:*
  - Difference of Gaussians in space and scale

---


Comparison of Keypoint Detectors

Table 7.1 Overview of feature 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>
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: e.g., Maximal Stable Extremal Regions

• Best choice often application dependent

• There have been extensive evaluations/comparisons
  – [Mikolajczyk et al., IJCV’05, PAMI’05]
  – All detectors/descriptors shown here work well
Feature descriptors

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

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

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

Lots of possibilities (this is a popular research area)

- Simple option: match square windows around the point
- State of the art approach: SIFT
Image representations

• Templates
  – Intensity, gradients, etc.

• Histograms
  – Color, texture, SIFT descriptors, etc.
Local Descriptors

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

• The ideal descriptor should be
  – Robust
  – Distinctive
  – Compact
  – Efficient

• Most available descriptors focus on edge/gradient information
  – Capture texture information

K. Grauman, B. Leibe
Image Representations: Histograms

Global histogram

- Represent distribution of features
  - Color, texture, depth, ...

Images from Dave Kauchak
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)
  - HOG: Histogram of Oriented Gradients
Orientation Normalization

- Compute orientation histogram
- Select dominant orientation
- Normalize: rotate to fixed orientation

[Lowe, SIFT, 1999]
Rotation invariance for feature descriptors

- Find dominant orientation of the image patch
  - This is given by $\mathbf{x}_{\text{max}}$, the eigenvector of $\mathbf{M}$ corresponding to $\lambda_{\text{max}}$ (the larger eigenvalue)
  - Rotate the patch according to this angle

Figure by Matthew Brown
Multiscale Oriented PatcheS descriptor

Take 40x40 square window around detected feature
- Scale to 1/5 size (using prefiltering)
- Rotate to horizontal
- Sample 8x8 square window centered at feature
- Intensity normalize the window by subtracting the mean, dividing by the standard deviation in the window

Adapted from slide by Matthew Brown
MOPS descriptor

• You can combine transformations together to get the final transformation

\[ T = ? \]
$T = M_{T_1}$
$T = M_R M_{T1}$
\[ T = M_S M_R M_{T1} \]
Translate

\[ T = M_{T_2}M_SM_RM_{T_1} \]
Crop
Detections at multiple scales

*Figure 1.* Multi-scale Oriented Patches (MOPS) extracted at five pyramid levels from one of the Matier images. The boxes show the feature orientation and the region from which the descriptor vector is sampled.
Invariance of MOPS

- Intensity
- Scale
- Rotation
What kind of things do we compute histograms of?

SIFT – Lowe IJCV 2004
### Scale Invariant Feature Transform

**Basic idea:**

- **DoG** for scale-space feature detection
- Take 16x16 square window around detected feature
  - Compute gradient orientation for each pixel
  - Throw out weak edges (threshold gradient magnitude)
  - Create histogram of surviving edge orientations

---

Adapted from slide by David Lowe
SIFT descriptor

Create histogram

• Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
• Compute an orientation histogram for each cell
• 16 cells * 8 orientations = 128 dimensional descriptor

Adapted from slide by David Lowe
**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
Ensure smoothness

• Trilinear interpolation
  – a given gradient contributes to 8 bins:
    4 in space times 2 in orientation
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

Image gradients

Keypoint descriptor
Properties of SIFT

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:
Other descriptors

• HOG: Histogram of Gradients (HOG)
  – Dalal/Triggs
  – Sliding window, pedestrian detection

• FREAK: Fast Retina Keypoint
  – Perceptually motivated
Summary

• Keypoint detection: repeatable and distinctive
  – Corners, blobs, stable regions
  – Harris, DoG

• Descriptors: robust and selective
  – spatial histograms of orientation
  – SIFT and variants are typically good for stitching and recognition
  – But, need not stick to one
Which features match?
Feature matching

Given a feature in $I_1$, how to find the best match in $I_2$?

1. Define distance function that compares two descriptors
2. Test all the features in $I_2$, find the one with min distance
Feature distance

How to define the difference between two features $f_1, f_2$?

- Simple approach: $L_2$ distance, $||f_1 - f_2||$
- can give good scores to ambiguous (incorrect) matches
Feature distance

How to define the difference between two features $f_1, f_2$?

- Better approach: ratio distance $= ||f_1 - f_2|| / ||f_1 - f_2'||$
  - $f_2$ is best SSD match to $f_1$ in $I_2$
  - $f_2'$ is 2nd best SSD match to $f_1$ in $I_2$
  - gives small values for ambiguous matches
PA2:
Feature detection and matching