Local Image Features

Computer Vision

James Hays

Read Szeliski 7.1.2 and 7.1.3

Acknowledgment: Many slides from Derek Hoiem and Grauman&Leibe 2008 AAAI Tutorial
“Flashed Face Distortion”
2nd Place in the 8th Annual
Best Illusion of the Year Contest, VSS 2012
Keep your eyes on the cross.
Project 2

The top 100 most confident local feature matches from a baseline implementation of project 2. In this case, 93 were correct (highlighted in green) and 7 were incorrect (highlighted in red).

Project 2: Local Feature Matching
This section: correspondence and alignment

- Correspondence: matching points, patches, edges, or regions across images
Overview of Keypoint Matching

1. Find a set of distinctive keypoints
2. Define a region around each keypoint
3. Extract and normalize the region content
4. Compute a local descriptor from the normalized region
5. Match local descriptors

\(d(f_A, f_B) < T\)
Review: Harris corner detector

• Define distinctiveness by local auto-correlation.
• Approximate local auto-correlation by second moment matrix
• Quantify distinctiveness (or cornerness) as function of the eigenvalues of the second moment matrix.
• But we don’t actually need to compute the eigenvalues by using the determinant and trace of the second moment matrix.
Harris Detector [Harris88]

- Second moment 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 (optionally, blur first)

\[
\det M = \lambda_1 \lambda_2
\]

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

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
Affine intensity change

\[ I \rightarrow a I + b \]

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

- Intensity scaling: \( I \rightarrow a I \)

Partially invariant to affine intensity change
Image translation

- Derivatives and window function are shift-invariant

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

Corner location is not covariant to scaling!

All points will be classified as edges.
Second moment ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner location is covariant w.r.t. rotation
Orientation Normalization

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

[Lowe, SIFT, 1999]
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

Harris

LoG

Hessian

MSER
Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding \( \mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}] \) each interest point.

3) Matching: Determine correspondence between descriptors in two views

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

• Templates
  – Intensity, gradients, etc.

• Histograms
  – Color, texture, SIFT descriptors, etc.
Image Representations: Histograms

Global histogram

- Represent distribution of features
  - Color, texture, depth, ...
Histogram: Probability or count of data in each bin

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

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

Images from Dave Kauchak
Image Representations: Histograms

Clustering

Use the same cluster centers for all images
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 histogram weighted by magnitude
• 8 orientations x 4x4 array = 128 dimensions
• Motivation: some sensitivity to spatial layout, but not too much.
Ensure smoothness

• Gaussian weight
• 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
Local Descriptors: SURF

Fast approximation of SIFT idea
Efficient computation by 2D box filters & integral images
\Rightarrow 6 \text{ times faster than SIFT}
Equivalent quality for object identification

GPU implementation available
Feature extraction @ 200Hz
(detector + descriptor, 640×480 img)
http://www.vision.ee.ethz.ch/~surf

[Bay, ECCV’06], [Cornelis, CVGPU’08]
Local Descriptors: Shape Context

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

Count = 4

Count = 10

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

Belongie & Malik, ICCV 2001
Shape Context Descriptor
Self-similarity Descriptor

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

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

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

Matching Local Self-Similarities across Images and Videos, Shechtman and Irani, 2007
Learning Local Image Descriptors, Winder and Brown, 2007

Image Patch → Smooth G(x, σ) → T-Block Filter → S-Block Pooling → N-Block Normalize → Descriptor

- 64x64 Pixels
- ~64x64 vectors of dimension k
- N histograms of dimension k

S1: SIFT grid with bilinear weights
S2: GLOH polar grid with bilinear radial and angular weights
S3: 3x3 grid with Gaussian weights
S4: 17 polar samples with Gaussian weights
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
  - Color rarely used
Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding $x_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$ each interest point.

3) Matching: Determine correspondence between descriptors in two views $x_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$.
Matching

• Simplest approach: Pick the nearest neighbor. Threshold on absolute distance
• Problem: Lots of self similarity in many photos
Distance: 0.34, 0.30, 0.40
Distance: 0.61
Distance: 1.22
Nearest Neighbor Distance Ratio

\[ \frac{NN_1}{NN_2} \]

where $NN_1$ is the distance to the first nearest neighbor and $NN_2$ is the distance to the second nearest neighbor.

- Sorting by this ratio (into ascending order) puts matches in order of confidence (in descending order of confidence).
Matching Local Features

- Nearest neighbor (Euclidean distance)
- Threshold ratio of nearest to 2\textsuperscript{nd} nearest descriptor
6.4 Matching to large databases

An important remaining issue for measuring the distinctiveness of features is how the reliability of matching varies as a function of the number of features in the database being matched. Most of the examples in this paper are generated using a database of 32 images with about 40,000 keypoints. Figure 10 shows how the matching reliability varies as a func-
SIFT Repeatability

Lowe IJCV 2004
SIFT Repeatability

Lowe IJCV 2004
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
## 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>

Tuytelaars Mikolajczyk 2008
Choosing a descriptor

• Again, need not stick to one

• For object instance recognition or stitching, SIFT or variant is a good choice
Things to remember

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

• Descriptors: robust and selective
  – spatial histograms of orientation
  – SIFT