Feature Matching and Robust Fitting

Computer Vision

James Hays

Read Szeliski 4.1

Acknowledgment: Many slides from Derek Hoiem and Grauman&Leibe 2008 AAAI Tutorial
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

\[ d(f_A, f_B) < T \]

K. Grauman, B. Leibe
Harris Corners – Why so complicated?

• Can’t we just check for regions with lots of gradients in the x and y directions?
  – No! A diagonal line would satisfy that criteria
Harris Detector [Harris88]

• Second moment matrix

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

1. Image derivatives (optionally, blur first)

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

2. Square of derivatives

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

4. Cornerness function – both eigenvalues are strong

\[
har = \det[\mu(\sigma_I, \sigma_D)] - \alpha[\trace(\mu(\sigma_I, \sigma_D))^2] = g(I_x^2)g(I_y^2) - [g(I_x I_y)]^2 - \alpha[g(I_x^2) + g(I_y^2)]^2
\]

5. Non-maxima suppression
Harris Corners – Why so complicated?

- What does the structure matrix look here?

\[
\begin{bmatrix}
C & -C \\
-C & C
\end{bmatrix}
\]
Harris Corners – Why so complicated?

- What does the structure matrix look here?

$$\begin{bmatrix} C & 0 \\ 0 & 0 \end{bmatrix}$$
Harris Corners – Why so complicated?

• What does the structure matrix look like here?

\[
\begin{bmatrix}
C & 0 \\
0 & C
\end{bmatrix}
\]
Review: Interest points

- Keypoint detection: repeatable and distinctive
  - Corners, blobs, stable regions
  - Harris, DoG, MSER
# 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>
Review: Local Descriptors

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

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

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

K. Grauman, B. Leibe
Feature Matching

- Simple criteria: One feature matches to another if those features are nearest neighbors and their distance is below some threshold.

- Problems:
  - Threshold is difficult to set
  - Non-distinctive features could have lots of close matches, only one of which is correct
How do we decide which features match?
Nearest Neighbor Distance Ratio

\[
\frac{NN1}{NN2}
\]

where \(NN1\) is the distance to the first nearest neighbor and \(NN2\) is the distance to the second nearest neighbor.

- Sorting by this ratio puts matches in order of confidence.
Matching Local Features

- Threshold based on the ratio of 1\textsuperscript{st} nearest neighbor to 2\textsuperscript{nd} nearest neighbor distance.

![Graph showing PDF for correct and incorrect matches.](Lowe IJCV 2004)
SIFT Repeatability

Matching location and scale
Matching location, scale, and orientation
Nearest descriptor in database

Repeatability (%) vs. Image noise

Lowe IJCV 2004
SIFT Repeatability
How do we decide which features match?
Fitting: find the parameters of a model that best fit the data

Alignment: find the parameters of the transformation that best align matched points
Fitting and Alignment

• Design challenges
  – Design a suitable goodness of fit measure
    • Similarity should reflect application goals
    • Encode robustness to outliers and noise
  – Design an optimization method
    • Avoid local optima
    • Find best parameters quickly
Fitting and Alignment: Methods

• Global optimization / Search for parameters
  – Least squares fit
  – Robust least squares
  – Iterative closest point (ICP)

• Hypothesize and test
  – Generalized Hough transform
  – RANSAC
Simple example: Fitting a line
Least squares line fitting

• Data: \((x_1, y_1), \ldots, (x_n, y_n)\)
• Line equation: \(y_i = mx_i + b\)
• Find \((m, b)\) to minimize

\[
E = \sum_{i=1}^{n} (y_i - mx_i - b)^2
\]

\[
E = \sum_{i=1}^{n} \left( \begin{bmatrix} x_i & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} - y_i \right)^2 = \left\| \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} - \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \right\|^2 = \|Ap - y\|^2
\]

\[
\]

\[
\frac{dE}{dp} = 2A^T Ap - 2A^T y = 0
\]

\[
A^T Ap = A^T y \Rightarrow p = (A^T A)^{-1} A^T y
\]

Matlab: \(p = A \backslash y;\)

Modified from S. Lazebnik
Least squares (global) optimization

Good
• Clearly specified objective
• Optimization is easy

Bad
• May not be what you want to optimize
• Sensitive to outliers
  – Bad matches, extra points
• Doesn’t allow you to get multiple good fits
  – Detecting multiple objects, lines, etc.
Least squares: Robustness to noise

• Least squares fit to the red points:
Least squares: Robustness to noise

• Least squares fit with an outlier:

Problem: squared error heavily penalizes outliers
Robust least squares (to deal with outliers)

General approach:

\[
\text{minimize } \sum_{i} \rho(u_i(x_i, \theta); \sigma)
\]

\[u^2 = \sum_{i=1}^{n} (y_i - mx_i - b)^2\]

\[u_i(x_i, \theta) \text{ – residual of } i\text{th point w.r.t. model parameters } \theta\]

\[\rho \text{ – robust function with scale parameter } \sigma\]

The robust function \(\rho\)

- Favors a configuration with small residuals
- Constant penalty for large residuals
Choosing the scale: Just right

The effect of the outlier is minimized
Choosing the scale: Too small

The error value is almost the same for every point and the fit is very poor
Choosing the scale: Too large

Behaves much the same as least squares
Robust estimation: Details

• Robust fitting is a nonlinear optimization problem that must be solved iteratively
• Least squares solution can be used for initialization
• Scale of robust function should be chosen adaptively based on median residual
Other ways to search for parameters (for when no closed form solution exists)

• Line search
  1. For each parameter, step through values and choose value that gives best fit
  2. Repeat (1) until no parameter changes

• Grid search
  1. Propose several sets of parameters, evenly sampled in the joint set
  2. Choose best (or top few) and sample joint parameters around the current best; repeat

• Gradient descent
  1. Provide initial position (e.g., random)
  2. Locally search for better parameters by following gradient
Hypothesize and test

1. Propose parameters
   – Try all possible
   – Each point votes for all consistent parameters
   – Repeatedly sample enough points to solve for parameters

2. Score the given parameters
   – Number of consistent points, possibly weighted by distance

3. Choose from among the set of parameters
   – Global or local maximum of scores

4. Possibly refine parameters using inliers
Hough Transform: Outline

1. Create a grid of parameter values

2. Each point votes for a set of parameters, incrementing those values in grid

3. Find maximum or local maxima in grid
Hough transform


Given a set of points, find the curve or line that explains the data points best

\[ y = m x + b \]

Hough space
Hough transform

Slide from S. Savarese
Hough transform


Issue: parameter space \([m,b]\) is unbounded...

Use a polar representation for the parameter space

$$x \cos \theta + y \sin \theta = \rho$$

Slide from S. Savarese
Hough transform - experiments

features

votes

Slide from S. Savarese
Hough transform - experiments

Noisy data

Need to adjust grid size or smooth

Slide from S. Savarese
Issue: spurious peaks due to uniform noise
1. Image $\rightarrow$ Canny
2. Canny $\rightarrow$ Hough votes
3. Hough votes $\rightarrow$ Edges

Find peaks and post-process
Hough transform example

http://ostatic.com/files/images/ss_hough.jpg
Incorporating image gradients

• Recall: when we detect an edge point, we also know its gradient direction

• But this means that the line is uniquely determined!

• Modified Hough transform:

  • For each edge point \((x,y)\)
    \[
    \theta = \text{gradient orientation at } (x,y) \\
    \rho = x \cos \theta + y \sin \theta \\
    H(\theta, \rho) = H(\theta, \rho) + 1
    \]
    end

\[
\nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]
\]

\[
\theta = \tan^{-1} \left( \frac{\partial f}{\partial y} / \frac{\partial f}{\partial x} \right)
\]
Finding lines using Hough transform

• Using m,b parameterization
• Using r, theta parameterization
  – Using oriented gradients
• Practical considerations
  – Bin size
  – Smoothing
  – Finding multiple lines
  – Finding line segments
Hough Transform

• How would we find circles?
  – Of fixed radius
  – Of unknown radius
  – Of unknown radius but with known edge orientation
Next lecture

• RANSAC

• Connecting model fitting with feature matching