Feature Matching and Robust Fitting

Computer Vision

CS 143, Brown

James Hays

Read Szeliski 4.1

Acknowledgment: Many slides from Derek Hoiem and Grauman&Leibe 2008 AAAI Tutorial
Project 2 questions?

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

CS 143: Introduction to Computer Vision

Brief

- Due: 11:59pm on Monday, October 7th, 2013
- Stencil code: /course/cs143/asgn/proj2/code/
- Data: /course/cs143/asgn/proj2/data/ includes 93 images from 9 different outdoor scenes.
- html writeup template: /course/cs143/asgn/proj2/html/
- Partial project materials are also available in proj2.zip (1.7 MB). Includes only the two test images shown above.
- Handin: cs143_handin_proj2
- Required files: README, code/, html/, html/index.html
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: Interest points

• Keypoint detection: repeatable and distinctive
  – Corners, blobs, stable regions
  – Harris, DoG, MSER
Review: Choosing an interest point 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
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
How do we decide which features match?
Feature Matching

• Szeliski 4.1.3
  – Simple feature-space methods
  – Evaluation methods
  – Acceleration methods
  – Geometric verification (Chapter 6)
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
Matching Local Features

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

![Graph showing SIFT repeatability](image-url)
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

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

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

\[
\]

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

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

Matlab: \(p = A \backslash y\);
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.
Robust least squares (to deal with outliers)

General approach:

minimize \[ \sum_i \rho(u_i(x_i, \theta); \sigma) \]

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

\[ u_i(x_i, \theta) \] – residual of \( i^{th} \) point w.r.t. model parameters \( \theta \)
\( \rho \) – robust function with scale parameter \( \sigma \)

The robust function \( \rho \)

- Favors a configuration with small residuals
- Constant penalty for large residuals
Robust Estimator

1. Initialize: e.g., choose $\theta$ by least squares fit and
   $\sigma = 1.5 \cdot \text{median}(\text{error})$

2. Choose params to minimize: $\sum_i \frac{\text{error}(\theta, \text{data}_i)^2}{\sigma^2 + \text{error}(\theta, \text{data}_i)^2}$
   - E.g., numerical optimization

3. Compute new $\sigma = 1.5 \cdot \text{median}(\text{error})$

4. Repeat (2) and (3) until convergence
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

\[ y = m \times x + b \]

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

\[ x \cos \theta + y \sin \theta = \rho \]
Hough transform - experiments

features

votes

Slide from S. Savarese
Hough transform - experiments

Noisy data

features

votes

Need to adjust grid size or smooth
Hough transform - experiments

Issue: spurious peaks due to uniform noise
1. Image → 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
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
Next lecture

• RANSAC

• Connecting model fitting with feature matching