Computer Vision - Lecture 13
Recognition with Local Features
15.12.2011

Bastian Leibe
RWTH Aachen
http://www.mmp.rwth-aachen.de/
leibe@umic.rwth-aachen.de

You Wanted A Script...
• We’ve created a script... for the part of the lecture on object recognition & categorization
  K. Grauman, B. Leibe
  Visual Object Recognition
  Morgan & Claypool publishers, 2011

• Chapter 3: Local Feature Extraction (Last 2 lectures)
• Chapter 4: Matching (Today’s topic)
• Chapter 5: Geometric Verification (Today’s topic)
  - Available on the L2P -

Course Outline
• Image Processing Basics
• Segmentation & Grouping
• Object Recognition
• Object Categorization I
  Sliding Window based Object Detection
• Local Features & Matching
  Local Features - Detection and Description
  Recognition with Local Features
  Indexing & Visual Vocabularies
• Object Categorization II
• 3D Reconstruction
• Motion and Tracking

Recap: Local Feature Matching Outline
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

Recap: Automatic Scale Selection
• Function responses for increasing scale (scale signature)

Recap: Laplacian-of-Gaussian (LoG)
• Interest points:
  Local maxima in scale space of Laplacian-of-Gaussian
  \( \Rightarrow \text{List of } (x, y, \sigma) \)
Recap: LoG Detector Responses

Recap: Key point localization with DoG
- Efficient implementation
  - Approximate LoG with a difference of Gaussians (DoG)
- Approach DoG Detector
  - Detect maxima of difference of Gaussian in scale space
  - Reject points with low contrast (threshold)
  - Eliminate edge responses

Recap: Harris-Laplace [Mikolajczyk '01]
1. Initialization: Multiscale Harris corner detection
2. Scale selection based on Laplacian
   (same procedure with Hessian ⇒ Hessian-Laplace)

Recap: SIFT Feature Descriptor
- Scale invariant Feature Transform
- Descriptor computation:
  - Divide patch into 4x4 sub-patches: 16 cells
  - Compute histogram of gradient orientations (8 reference angles)
  - For all pixels inside each sub-patch
  - Resulting descriptor: $4 \times 4 \times 8 = 128$ dimensions

Topics of This Lecture
- Recognition with Local Features
  - Matching local features
  - Finding consistent configurations
  - Alignment: linear transformations
  - Affine estimation
  - Homography estimation
- Dealing with Outliers
  - RANSAC
  - Generalized Hough Transform
- Indexing with Local Features
  - Inverted file index
  - Visual Words
  - Visual Vocabulary construction
  - $\text{tf-idf}$ weighting

Recognition with Local Features
- Image content is transformed into local features that are invariant to translation, rotation, and scale
- Goal: Verify if they belong to a consistent configuration
**Concepts: Warping vs. Alignment**

- **Warping**: Given a source image and a transformation, what does the transformed output look like?
- **Alignment**: Given two images with corresponding features, what is the transformation between them?

**Parametric (Global) Warping**

Transformation $T$ is a coordinate-changing machine:

$$T(p) = p'$$

- What does it mean that $T$ is global?
  - It’s the same for any point $p$
  - It can be described by just a few numbers (parameters)
- Let’s represent $T$ as a matrix:

$$p' = Mp,$$

where $M$ is a $2 	imes 2$ matrix.

**What Can be Represented by a 2x2 Matrix?**

- **2D Scaling?**
  
  $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- **2D Rotation around (0,0)?**

  $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- **2D Shearing?**

  $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & sh_y \\ sh_x & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

**2D Linear Transforms**

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- Only linear 2D transformations can be represented with a 2x2 matrix.
- Linear transformations are combinations of...
  - Scale, Rotation, Shear, and Mirror

**Homogeneous Coordinates**

- **Q**: How can we represent translation as a 3x3 matrix using homogeneous coordinates?

  $$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$
Basic 2D Transformations

- Basic 2D transformations as 3x3 matrices

\[
\begin{bmatrix}
    x' \\
    y' \\
    1
\end{bmatrix} =
\begin{bmatrix}
    a & b & c \\
    d & e & f \\
    g & h & i
\end{bmatrix}
\begin{bmatrix}
    x \\
    y \\
    1
\end{bmatrix}
\]

- **Rotation**
  \[
  \begin{bmatrix}
  x' \\
  y' \\
  1
\end{bmatrix} =
\begin{bmatrix}
  \cos \theta & -\sin \theta & 0 \\
  \sin \theta & \cos \theta & 0 \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}
\]

- **Translation**
  \[
  \begin{bmatrix}
  x' \\
  y' \\
  1
\end{bmatrix} =
\begin{bmatrix}
  1 & 0 & t_x \\
  0 & 1 & t_y \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}
\]

- **Scaling**
  \[
  \begin{bmatrix}
  x' \\
  y' \\
  1
\end{bmatrix} =
\begin{bmatrix}
  a & 0 & 0 \\
  0 & b & 0 \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}
\]

- **Shearing**
  \[
  \begin{bmatrix}
  x' \\
  y' \\
  1
\end{bmatrix} =
\begin{bmatrix}
  1 & s_x & 0 \\
  s_y & 1 & 0 \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}
\]

2D Affine Transformations

- Affine transformations are combinations of...
  - Linear transformations, and
  - Translations

- **Parallel lines remain parallel**

### Projective Transformations

\[
\begin{bmatrix}
  x' \\
  y' \\
  w
\end{bmatrix} =
\begin{bmatrix}
  a & b & c \\
  d & e & f \\
  g & h & i
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  w
\end{bmatrix}
\]

- **Projective transformations:**
  - Affine transformations, and
  - Projective warps

- **Parallel lines do not necessarily remain parallel**

#### Alignment Problem

- We have previously considered how to fit a model to image evidence
  - e.g., a line to edge points

- In alignment, we will fit the parameters of some transformation according to a set of matching feature pairs (“correspondences”).

#### Let’s Start with Affine Transformations

- Simple fitting procedure (linear least squares)
- Approximates viewpoint changes for roughly planar objects and roughly orthographic cameras
- Can be used to initialize fitting for more complex models

#### Fitting an Affine Transformation

- Affine model approximates perspective projection of planar objects
**Fitting an Affine Transformation**

- Assuming we know the correspondences, how do we get the transformation?

\[
\begin{bmatrix}
  x' \\
  y'
\end{bmatrix} =
\begin{bmatrix}
  m_1 & m_2 \\
  m_3 & m_4
\end{bmatrix}
\begin{bmatrix}
  x \\
  y
\end{bmatrix} +
\begin{bmatrix}
  t_1 \\
  t_2
\end{bmatrix}
\]

**Recall: Least Squares Estimation**

- Set of data points: \((X_1, X_2, X_3, X_4)\)
- Goal: a linear function to predict \(X'\) from \(X\):
  \[X' = AX + B\]
- We want to find \(a\) and \(b\).
- How many \((X, X')\) pairs do we need?
  \[X_1a + b = X_1'\]
  \[X_2a + b = X_2'\]
  \[
  \begin{bmatrix}
    X_1 & 1 & a & b \\
    X_2 & 1 & a & b
  \end{bmatrix}
  =
  \begin{bmatrix}
    X_1' \\
    X_2'
  \end{bmatrix}
  \]

**Homography**

- A projective transform is a mapping between any two perspective projections with the same center of projection.
  - i.e. two planes in 3D along the same sight ray
- Properties
  - Rectangle should map to arbitrary quadrilateral
  - Parallel lines aren’t but must preserve straight lines
- This is called a homography

\[
\begin{bmatrix}
  x' \\
  y' \\
  1
\end{bmatrix} =
\begin{bmatrix}
  h_1 & h_2 & h_3 & x \\
  h_4 & h_5 & h_6 & y \\
  h_7 & h_8 & h_9 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}
\]

Set scale factor to 1 ⇒ 8 parameters left.

Slide credit: Kristen Grauman

Slide adapted from Alexej Efros

B. Leibe
Fitting a Homography

- Estimating the transformation

\[ x' = Hx \]

\[ x'' = \frac{1}{x'} x' \]

\[ y' = \frac{y}{x'} \]

\[ y'' = \frac{y'}{x'} \]

- Image coordinates

- Homogenous coordinates

- Matrix notation

Slide credit: Krystian Mikolajczyk

B. Leibe

Computer Vision WS 11/12
Fitting a Homography

- Estimating the transformation

\[
\begin{pmatrix}
1 & a_1 & x_1 \\
1 & a_2 & x_2 \\
\vdots & \vdots & \vdots \\
1 & a_n & x_n \\
\end{pmatrix}
\begin{pmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
\vdots & \vdots & \vdots \\
0 & 0 & 1 \\
\end{pmatrix}
\begin{pmatrix}
y_1 \\
y_2 \\
\vdots \\
y_n \\
\end{pmatrix}
= \begin{pmatrix}
x_1 \\
x_2 \\
\vdots \\
x_n \\
\end{pmatrix}
\]

\[Ax = b\]

- Solution:
  - Null-space vector of \( A \)
  - Corresponds to smallest singular vector

\[ A \sim UDV \]

Estimating the transformation

\[ Ah = 0 \]

Minimizes least square error

Image Warping with Homographies

Uses: Analyzing Patterns and Shapes

- What is the shape of the b/w floor pattern?

Fitting a Homography

- Estimating the transformation

\[ Ah = 0 \]

Solution:

- Null-space vector of \( A \)

\[ A = U \Sigma V^T \]

Minimizes least square error

Analyzing Patterns and Shapes

From Martin Kemp: *The Science of Art* (manual reconstruction)
Topics of This Lecture

- Recognition with Local Features
  - Matching local features
  - Finding consistent configurations
  - Alignment: linear transformations
  - Affine estimation
  - Homography estimation
- Dealing with Outliers
  - RANSAC
  - Generalized Hough Transform
- Indexing with Local Features
  - Inverted file index
  - Visual Words
  - Visual Vocabulary construction
  - tf-idf weighting

Problem: Outliers

- Outliers can hurt the quality of our parameter estimates, e.g.,
  - An erroneous pair of matching points from two images
  - A feature point that is noise or doesn’t belong to the transformation we are fitting.

Example: Least-Squares Line Fitting

- Assuming all the points that belong to a particular line are known

Outliers Affect Least-Squares Fit

Strategy 1: RANSAC [Fischler81]

- RANdom SAmple Consensus
  - Approach: we want to avoid the impact of outliers, so let’s look for “inliers”, and use only those.
  - Intuition: if an outlier is chosen to compute the current fit, then the resulting line won’t have much support from rest of the points.
RANSAC

RANSAC loop:
1. Randomly select a seed group of points on which to base transformation estimate (e.g., a group of matches)
2. Compute transformation from seed group
3. Find inliers to this transformation
4. If the number of inliers is sufficiently large, recompute least-squares estimate of transformation on all of the inliers
   • Keep the transformation with the largest number of inliers

RANSAC Line Fitting Example

• Task: Estimate the best line
   How many points do we need to estimate the line?

Sample two points

Fit a line to them

Total number of points within a threshold of line.

"7 inlier points"
RANSAC Line Fitting Example

• Task: Estimate the best line

Repeat, until we get a good result.

RANSAC: How many samples?

• How many samples are needed?
  - Suppose $w$ is fraction of inliers (points from line).
  - $n$ points needed to define hypothesis (2 for lines)
  - $k$ samples chosen.

• Prob. that a single sample of $n$ points is correct: $w^n$

• Prob. that all $k$ samples fail is: $(1 - w^k)^k$

⇒ Choose $k$ high enough to keep this below desired failure rate.

After RANSAC

• RANSAC divides data into inliers and outliers and yields estimate computed from minimal set of inliers.

• Improve this initial estimate with estimation over all inliers (e.g. with standard least-squares minimization).

• But this may change inliers, so alternate fitting with re-classification as inlier/outlier.

Example: Finding Feature Matches

• Find best stereo match within a square search window (here 300 pixels$^2$)

• Global transformation model: epipolar geometry
Example: Finding Feature Matches
- Find best stereo match within a square search window (here 300 pixels x 300 pixels).
- Global transformation model: epipolar geometry

Before RANSAC

After RANSAC

Problem with RANSAC
- In many practical situations, the percentage of outliers (incorrect putative matches) is often very high (90% or above).
- Alternative strategy: Generalized Hough Transform

Strategy 2: Generalized Hough Transform
- Suppose our features are scale- and rotation-invariant
  - Then a single feature match provides an alignment hypothesis (translation, scale, orientation).

Pose Clustering and Verification with SIFT
- To detect instances of objects from a model base:
  1. Index descriptors
     - Distinctive features narrow down possible matches

Indexing Local Features
- Model base
- New image
Pose Clustering and Verification with SIFT

- To detect instances of objects from a model base:
  1. Index descriptors
     - Distinctive features narrow down possible matches
  2. Generalized Hough transform to vote for poses
     - Keypoints have record of parameters relative to model coordinate system
  3. Affine fit to check for agreement between model and image features
     - Fit and verify using features from Hough bins with 3+ votes

Object Recognition Results

- Background subtract for model boundaries
- Objects recognized
- Recognition in spite of occlusion

Recall: Difficulties of Voting

- Noise/clutter can lead to as many votes as true target.
- Bin size for the accumulator array must be chosen carefully.
- (Recall Hough Transform)
- In practice, good idea to make broad bins and spread votes to nearby bins, since verification stage can prune bad vote peaks.

Location Recognition

Training

Applications: Specific Object Recognition

- Sony Aibo
  (Evolution Robotics)
- SIFT usage
  - Recognize docking station
  - Communicate with visual cards

Summary

- Recognition by alignment: looking for object and pose that fits well with image
  - Use good correspondences to designate hypotheses.
  - Invariant local features offer more reliable matches.
  - Find consistent "inlier" configurations in clutter
    - Generalized Hough Transform
    - RANSAC
- Alignment approach to recognition can be effective if we find reliable features within clutter.
  - Application: large-scale image retrieval
  - Application: recognition of specific (mostly planar) objects
    - Movie posters
    - Books
    - CD covers
Topics of This Lecture

- Recognition with Local Features
  - Matching local features
  - Finding consistent configurations
  - Alignment: linear transformations
  - Affine estimation
  - Homography estimation

- Dealing with Outliers
  - RANSAC
  - Generalized Hough Transform

- Indexing with Local Features
  - Inverted file index
  - Visual Words
  - Visual Vocabulary construction
  - tf-idf weighting

Application: Mobile Visual Search

- Take photos of objects as queries for visual search

Large-Scale Image Matching Problem

- How can we perform this matching step efficiently?

Indexing Local Features

- Each patch / region has a descriptor, which is a point in some high-dimensional feature space (e.g., SIFT)

- When we see close points in feature space, we have similar descriptors, which indicates similar local content.

- This is of interest for many applications
  - E.g. Image matching,
  - E.g. Retrieving images of similar objects,
  - E.g. Object recognition, categorization, 3d Reconstruction,...
Indexing Local Features

- With potentially thousands of features per image, and hundreds to millions of images to search, how to efficiently find those that are relevant to a new image?
- Low-dimensional descriptors (e.g. through PCA):
  - Can use standard efficient data structures for nearest neighbor search
- High-dimensional descriptors
  - Approximate nearest neighbor search methods more practical
  - Inverted file indexing schemes

Indexing Local Features: Inverted File Index

- For text documents, an efficient way to find all pages on which a word occurs is to use an index...
- We want to find all images in which a feature occurs.
- To use this idea, we’ll need to map our features to “visual words”.

Text Retrieval vs. Image Search

- What makes the problems similar, different?

Visual Words: Main Idea

- Extract some local features from a number of images...

Visual Words: Main Idea
**Visual Words: Main Idea**

Each point is a local descriptor, e.g. SIFT vector.

**Indexing with Visual Words**

Map high-dimensional descriptors to tokens/words by quantizing the feature space.

- Quantize via clustering, let cluster centers be the prototype “words”

**Visual Words**

- Example: each group of patches belongs to the same visual word
**Visual Words: Texture Representation**

- First explored for texture and material representations.
- **Texton** = cluster center of filter responses over collection of images.
- Describe textures and materials based on distribution of prototypical texture elements.


**Visual Words**

- More recently used for describing scenes and objects for the sake of indexing or classification.

Sivic & Zisserman 2003; Csurka, Bray, Dance, & Fan 2004; many others.

**Inverted File for Images of Visual Words**

When will this give us a significant gain in efficiency?

**Visual Vocabulary Formation**

Design choices:
- Sampling strategy: where to extract features?
- Clustering / quantization algorithm
- Unsupervised vs. supervised
- What corpus provides features (universal vocabulary?)
- Vocabulary size, number of words

**Sampling Strategies**

- Sparse, at interest points
- Dense, uniformly
- Randomly

- To find specific, textured objects, sparse sampling from interest points often more reliable.
- Multiple complementary interest operators offer more image coverage.
- For object categorization, dense sampling offers better coverage.

[See Nowak, Jurie & Triggs, ECCV 2006]
Clustering / Quantization Methods

- k-means (typical choice), agglomerative clustering, mean-shift, ...

- Hierarchical clustering: allows faster insertion / word assignment while still allowing large vocabularies
  - Vocabulary tree [Nister & Stewenius, CVPR 2006]

Example: Recognition with Vocabulary Tree

- Tree construction:

Vocabulary Tree

- Training: Filling the tree

Vocabulary Tree

- Training: Filling the tree

Vocabulary Tree

- Training: Filling the tree

Vocabulary Tree

- Training: Filling the tree
Vocabulary Tree

• Training: Filling the tree

Vocabulary Tree

• Recognition

Quiz Questions

• What is the computational advantage of the hierarchical representation vs. a flat vocabulary?

• What dangers does such a representation carry?

Vocabulary Tree: Performance

• Evaluated on large databases
  - Indexing with up to 1M images
  - Online recognition for database of 50,000 CD covers
    - Retrieval in ~1s

  - Experimental finding that large vocabularies can be beneficial for recognition

[Nister & Stewenius, CVPR’06]

Vocabulary Size

• Larger vocabularies can be advantageous...
  - But what happens when the vocabulary gets too large?
    - Efficiency?
    - Robustness?

tf-idf Weighting

• Term frequency - inverse document frequency
• Describe frame by frequency of each word within it, downweight words that appear often in the database
• (Standard weighting for text retrieval)

\[ t_i = \frac{n_{id}}{n_i} \log \frac{N}{n_i} \]

- Number of occurrences of word \( i \) in document \( d \)
- Number of occurrences of word \( i \) in whole database

- Total number of documents in database
- Number of words in document \( d \)
Summary: Indexing features

- Detect or sample features
- List of positions, scales, orientations
- Describe features
- Associated list of d-dimensional descriptors
- Index each one into pool of descriptors from previously seen images
- Quantize to form “bag of words” vector for the image

References and Further Reading

- A detailed description of local feature extraction and recognition can be found in Chapters 3-5 of Grauman & Leibe (available on the L2P).
- More details on RANSAC can also be found in Chapter 4.7 of Hartley & Zisserman.

Summary

- Local invariant features
  - Distinctive matches possible in spite of significant view change, useful not only to provide matches for image stitching/multi-view geometry, but also to find objects and scenes.
  - To find correspondences among detected features, measure distance between descriptors, and look for most similar patches.

- Visual vocabulary representation
  - Quantize feature space to make discrete set of visual words
  - Index individual words
  - Inverted index: pre-compute index to enable faster search at query time

- Geometric verification
  - Use RANSAC to estimate transformation between feature constellations in both images

References:

- K. Grauman, B. Leibe
  Visual Object Recognition
  Morgan & Claypool publishers, 2011

- R. Hartley, A. Zisserman
  Multiple View Geometry in Computer Vision
  2nd Ed., Cambridge Univ. Press, 2004