Instance-level recognition – part 2

Josef Sivic
http://www.di.ens.fr/~josef
INRIA, WILLOW, ENS/INRIA/CNRS UMR 8548
Laboratoire d’Informatique, Ecole Normale Supérieure, Paris

Outline

1. Local invariant features (C. Schmid)

2. Matching and recognition with local features (J. Sivic)

3. Efficient visual search (J. Sivic)

4. Very large scale visual indexing – recent work (C. Schmid)

Practical session – Instance-level recognition and search
[Try your wifi network access.]
Image matching and recognition with local features

The goal: establish **correspondence** between two or more images

Image points $x$ and $x'$ are **in correspondence** if they are projections of the same 3D scene point $X$.

Images courtesy A. Zisserman
Example I: **Wide baseline matching and 3D reconstruction**

Establish correspondence between two (or more) images.

[Schaffalitzky and Zisserman ECCV 2002]
Example I: Wide baseline matching and 3D reconstruction
Establish correspondence between two (or more) images.

[Schaffalitzky and Zisserman ECCV 2002]
Example II: Object recognition

Establish correspondence between the target image and (multiple) images in the model database.

[D. Lowe, 1999]
Example III: Visual search

Given a query image, find images depicting the same place/object in a large unordered image collection.

Find these landmarks...in these images and 1M more
Establish correspondence between the query image and all images from the database depicting the same object / scene.
Why is it difficult?

Want to establish correspondence despite possibly large changes in scale, viewpoint, lighting and partial occlusion … and the image collection can be very large (e.g. 1M images)
Approach

Pre-processing (so far):
• Detect local features.
• Extract descriptor for each feature.

Matching:
1. Establish tentative (putative) correspondences based on local appearance of individual features (their descriptors).
2. Verify matches based on semi-local / global geometric relations.
Example I: Two images - “Where is the Graffiti?”
Step 1. Establish tentative correspondence

Establish tentative correspondences between object model image and target image by nearest neighbour matching on SIFT vectors.

Need to solve some variant of the “nearest neighbor problem” for all feature vectors, \( x_j \in \mathcal{R}^{128} \), in the query image:

\[
\forall j \quad NN(j) = \arg \min_i ||x_i - x_j||,
\]

where, \( x_i \in \mathcal{R}^{128} \), are features in the target image.

Can take a long time if many target images are considered (see later).
Step 1. Establish tentative correspondence

Establish tentative correspondences between object model image and target image by nearest neighbour matching on SIFT vectors

Need to solve some variant of the “nearest neighbor problem” for all feature vectors, $x_j \in \mathcal{R}^{128}$, in the query image:

$$\forall j \quad NN(j) = \arg \min_i ||x_i - x_j||,$$

where, $x_i \in \mathcal{R}^{128}$, are features in the target image.

Can take a long time if many target images are considered (see later).
Problem with matching on local descriptors alone

• too much individual invariance
• each region can affine deform independently (by different amounts)
• locally, appearance can be ambiguous

Solution: use semi-local and global spatial relations to verify matches.
Initial matches

Nearest-neighbor search based on appearance descriptors alone.

After spatial verification

Example I: Two images - “Where is the Graffiti?”
Step 2: Spatial verification

1. Semi-local constraints
   Constraints on spatially close-by matches

2. Global geometric relations
   Require a consistent global relationship between all matches
Semi-local constraints: Example I. – neighbourhood consensus

Fig. 4. Semi-local constraints: neighbours of the point have to match and angles have to correspond. Note that not all neighbours have to be matched correctly.

[Schmid&Mohr, PAMI 1997]
Semi-local constraints:
Example I. –
neighbourhood consensus

[Schaffalitzky & Zisserman, CIVR 2004]
Semi-local constraints: Example II.

Figure 5: Surface contiguity filter. a) the pattern of intersection between neighboring correct region matches is preserved by transformations between the model and the test images, because the surface is contiguous and smooth. b) the filter evaluates this property by testing the conservation of the area ratios.

[Ferrari et al., IJCV 2005]
Geometric verification with global constraints

- All matches must be consistent with a global geometric relation / transformation.

- Need to simultaneously (i) estimate the geometric relation / transformation and (ii) the set of consistent matches.
Examples of global constraints

1 view and known 3D model.
• Consistency with a (known) 3D model.

2 views
• Epipolar constraint
• 2D transformations
  • Similarity transformation
  • Affine transformation
  • Projective transformation

N-views
Are all images consistent with a single 3D model?
3D constraint: example (not considered here)

- Matches must be consistent with a 3D model

Offline: Build a 3D model

3 (out of 20) images used to build the 3D model

 Recovered 3D model

[Lazebnik, Rothganger, Schmid, Ponce, CVPR'03]
3D constraint: example (not considered here)

• Matches must be consistent with a 3D model

Offline: Build a 3D model

3 (out of 20) images used to build the 3D model

At test time:

Object recognized in a previously unseen pose

Recovered 3D model

Recovered pose

[Lazebnik, Rothganger, Schmid, Ponce, CVPR’03]
3D constraint: example (not considered here)

With a given 3D model (set of known 3D points $X$’s) and a set of measured 2D image points $x$, the goal is to find camera matrix $P$ and a set of geometrically consistent correspondences $x \leftrightarrow X$. 

$x = PX$

$P : 3 \times 4$ matrix
$x : 4$-vector
$x : 3$-vector
Epipolar geometry (not considered here)

In general, two views of a 3D scene are related by the epipolar constraint.

- A point in one view “generates” an epipolar line in the other view
- The corresponding point lies on this line.

Slide credit: A. Zisserman
2D transformation models

_similarity (translation, scale, rotation)

_Affine

_Projective (homography)
Planes in the scene induce homographies

Points on the plane transform as $x' = H x$, where $x$ and $x'$ are image points (in homogeneous coordinates), and $H$ is a 3x3 matrix.
Case II: Cameras rotating about their centre

- The two image planes are related by a homography $H$
- $H$ depends only on the relation between the image planes and camera centre, $C$, not on the 3D structure
Homography is often approximated well by 2D affine geometric transformation
Homography is often approximated well by 2D affine geometric transformation – Example II.

Two images with similar camera viewpoint

Tentative matches

Matches consistent with an affine transformation
Example: estimating 2D affine transformation

- 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
Example: estimating 2D affine transformation

- 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

Assume we know the correspondences, how do we get the transformation?

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

\[
\begin{bmatrix}
    x_i & y_i & 0 & 0 & 1 & 0 \\
    0 & 0 & x_i & y_i & 0 & 1 \\
    \vdots & \vdots & \vdots & \vdots & \vdots \\
    m_1 & m_2 & m_3 & m_4 & t_1 & t_2 \\
\end{bmatrix}
= \begin{bmatrix}
    \cdots \\
    x'_i \\
    y'_i \\
    \cdots \\
\end{bmatrix}
\]
Fitting an affine transformation

$$\begin{bmatrix}
  x_i & y_i & 0 & 0 & 1 & 0 \\
  0 & 0 & x_i & y_i & 0 & 1 \\
  \vdots & \vdots & \vdots & \vdots & \vdots & \vdots
\end{bmatrix} \begin{bmatrix}
  m_1 \\
  m_2 \\
  m_3 \\
  m_4 \\
  t_1 \\
  t_2
\end{bmatrix} = \begin{bmatrix}
  \cdots \\
  x'_i \\
  y'_i \\
  \cdots
\end{bmatrix}$$

Linear system with six unknowns
Each match gives us two linearly independent equations: need at least three to solve for the transformation parameters
Dealing with outliers

The set of putative matches may contain a high percentage (e.g. 90%) of outliers

How do we fit a geometric transformation to a small subset of all possible matches?

Possible strategies:
  • **RANSAC**
  • Hough transform
Example: Robust line estimation - RANSAC

Fit a line to 2D data containing outliers

There are two problems:
1. a line fit which minimizes perpendicular distance
2. a classification into inliers (valid points) and outliers

Solution: use robust statistical estimation algorithm RANSAC (RANdom Sample Consensus) [Fishler & Bolles, 1981]
RANSAC robust line estimation

Repeat

1. Select random sample of 2 points
2. Compute the line through these points
3. Measure support (number of points within threshold distance of the line)

Choose the line with the largest number of inliers

- Compute least squares fit of line to inliers (regression)
Algorithm summary – RANSAC robust estimation of 2D affine transformation

Repeat

1. Select 3 point to point correspondences
2. Compute $H$ (2x2 matrix) + $t$ (2x1) vector for translation
3. Measure support (number of inliers within threshold distance, i.e. $d_{\text{transfer}}^2 < t$)
   \[
   d_{\text{transfer}}^2 = d(x, H^{-1}x')^2 + d(x', Hx)^2
   \]

Choose the $(H,t)$ with the largest number of inliers
(Re-estimate $(H,t)$ from all inliers)
How many samples?

**Number of samples** \( N \)

- Choose \( N \) so that, with probability \( p \), at least one random sample is free from outliers
- e.g.:
  - \( p = 0.99 \)
  - outlier ratio: \( e \)

Probability a randomly picked point is an inlier

\[
\left( 1 - \left( 1 - e \right)^s \right)^N = 1 - p
\]

Probability of all points in a sample (of size \( s \)) are inliers

Source: M. Pollefeys
Number of samples $N$

- Choose $N$ so that, with probability $p$, at least one random sample is free from outliers
- e.g.:  
  > $p=0.99$
  > outlier ratio: $e$

$$
\left(1 - (1 - e)^s\right)^N = 1 - p
$$

Probability that all $N$ samples (of size $s$) are corrupted (contain an outlier)

$$
N = \log(1 - p)/\log(1 - (1 - e)^s)
$$

<table>
<thead>
<tr>
<th>$s$</th>
<th>5%</th>
<th>10%</th>
<th>20%</th>
<th>30%</th>
<th>40%</th>
<th>50%</th>
<th>90%</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>43</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>11</td>
<td>17</td>
<td>458</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>4</td>
<td>7</td>
<td>11</td>
<td>19</td>
<td>35</td>
<td>4603</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>17</td>
<td>34</td>
<td>72</td>
<td>4.6e4</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>6</td>
<td>12</td>
<td>26</td>
<td>57</td>
<td>146</td>
<td>4.6e5</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>7</td>
<td>16</td>
<td>37</td>
<td>97</td>
<td>293</td>
<td>4.6e6</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>8</td>
<td>20</td>
<td>54</td>
<td>163</td>
<td>588</td>
<td>4.6e7</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>9</td>
<td>26</td>
<td>78</td>
<td>272</td>
<td>1177</td>
<td>4.6e8</td>
</tr>
</tbody>
</table>

Source: M. Pollefeys
How to reduce the number of samples needed?

1. Reduce the proportion of outliers.
2. Reduce the sample size
   - use simpler model (e.g. similarity instead of affine tnf.)
   - use local information (e.g. a region to region correspondence is equivalent to (up to) 3 point to point correspondences).

<table>
<thead>
<tr>
<th>Number of samples $N$</th>
<th>proportion of outliers $\varepsilon$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5%</td>
</tr>
<tr>
<td>$s$</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
</tr>
</tbody>
</table>
Example: restricted affine transform

1. Test each correspondence
Example: restricted affine transform

2. Compute a (restricted) planar affine transformation (5 dof)

Need just one correspondence
Example: restricted affine transform

3. Score by number of consistent matches

Re-estimate full affine transformation (6 dof)
Example II: (see practical later today)

**Similarity transformation** is specified by four parameters: scale factor $s$, rotation $\theta$, and translations $t_x$ and $t_y$.

$$
\begin{bmatrix}
x' \\
y'
\end{bmatrix} = s R(\theta) \begin{bmatrix}
x \\
y
\end{bmatrix} + \begin{bmatrix}
t_x \\
t_y
\end{bmatrix}
$$

Recall, each SIFT detection has: position $(x_i, y_i)$, scale $s_i$, and orientation $\theta_i$.

How many correspondences are needed to compute similarity transformation?
RANSAC (references)


Extensions:

B. Tordoff and D. Murray, “Guided Sampling and Consensus for Motion Estimation, ECCV’03

D. Nister, “Preemptive RANSAC for Live Structure and Motion Estimation, ICCV’03

Chum, O.; Matas, J. and Obdrzalek, S.: Enhancing RANSAC by Generalized Model Optimization, ACCV’04

Chum, O.; and Matas, J.: Matching with PROSAC - Progressive Sample Consensus, CVPR 2005

Philbin, J., Chum, O., Isard, M., Sivic, J. and Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching, CVPR’07

Chum, O. and Matas. J.: Optimal Randomized RANSAC, PAMI’08
Outline

1. Local invariant features (C. Schmid)

2. Matching and recognition with local features (J. Sivic)

3. Efficient visual search (J. Sivic)

4. Very large scale visual indexing – recent work (C. Schmid)

Practical session – Instance-level recognition and search