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: 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: 4x4x8 = 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
  - 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

Local Features, e.g. SIFT

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:
  
  $p' = T(p)$

- 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:

  $P' = MP$,
  
  $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} M$

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**?

  $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & sh_x \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$

- **2D Translation**?

  $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + t$

  NO!
**Homogeneous Coordinates**

- Q: How can we represent translation as a 3x3 matrix using homogeneous coordinates?
  
  \[ x' = x + t_x \]
  
  \[ y' = y + t_y \]

- A: Using the rightmost column:

  \[
  \begin{bmatrix}
  1 & 0 & t_x \\
  0 & 1 & t_y \\
  0 & 0 & 1
  \end{bmatrix}
  \]

**Basic 2D Transformations**

- Basic 2D transformations as 3x3 matrices

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

  **Translation**

  \[
  \begin{bmatrix}
  x' \\
  y' \\
  1
  \end{bmatrix}
  =
  \begin{bmatrix}
  1 & 0 & 0 \\
  0 & 1 & 0 \\
  0 & 0 & 1
  \end{bmatrix}
  \begin{bmatrix}
  x \\
  y \\
  1
  \end{bmatrix}
  \]

  **Scaling**

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

  **Rotation**

  \[
  \begin{bmatrix}
  x' \\
  y' \\
  1
  \end{bmatrix}
  =
  \begin{bmatrix}
  
  \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**

\[
\begin{bmatrix}
  x' \\
  y' \\
  w'
\end{bmatrix}
= \begin{bmatrix}
  a & b & c \\
  d & e & f \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  w
\end{bmatrix}
\]

- *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

\[ Ax + b = X' \]

Assuming we know the correspondences, how do we get the transformation? Once we have solved for the parameters, how do we compute the coordinates of the corresponding point for \((x, y)\)?

Recall: Least Squares Estimation

- Set of data points: \((X_i, X'_i)\), \((X_j, X'_j)\), \((X_k, X'_k)\)
- Goal: a linear function to predict \(X'\)'s from \(X\):

\[ X' = Ax + b \]

We want to find \(a\) and \(b\).

- How many \((X, X')\) pairs do we need?

\[
\begin{bmatrix}
X_1 & 1 & a \\
X_2 & 1 & b \\
\vdots & \vdots & \vdots \\
X_i & 1 & a \\
\end{bmatrix}
\begin{bmatrix}
X'_1 \\
X'_2 \\
\vdots \\
X'_i \\
\end{bmatrix}
= \begin{bmatrix}
X'_1 \\
X'_2 \\
\vdots \\
X'_i \\
\end{bmatrix}
\]

\[ Ax = B \]

What if the data is noisy?

Overconstrained problem: \(\min \| Ax - B \|^2\)

Solution: \(x = A^+ B\)

Matlab: \(x = A \backslash B\)

Fitting an Affine Transformation

- How many matches (correspondence pairs) do we need to solve for the transformation parameters?
- Once we have solved for the parameters, how do we compute the coordinates of the corresponding point for \((x, y)\)?

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 parallel
  - but must preserve straight lines
- This is called a homography.
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

\[
\begin{bmatrix}
wx' \\
wz' \\
w'
p'
\end{bmatrix} =
\begin{bmatrix}
h_{11} & h_{12} & h_{13} & x \\
h_{21} & h_{22} & h_{23} & y \\
h_{31} & h_{32} & h_{33} & 1
\end{bmatrix}
\begin{bmatrix}
w \\
1
\end{bmatrix}
\]

Set scale factor to 1 ⇒ 8 parameters left.

Fitting a Homography

- Estimating the transformation

\[
\begin{bmatrix}
x' \\
y' \\
z'
\end{bmatrix} =
\begin{bmatrix}
h_{11} & h_{12} & h_{13} & x \\
h_{21} & h_{22} & h_{23} & y \\
h_{31} & h_{32} & h_{33} & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix}
\]

\[
x' = Hx
\]

\[
x'' = \frac{1}{w} x'
\]
Estimating the transformation

$A \leftrightarrow x \leftrightarrow x_B \leftrightarrow x_A \leftrightarrow x_B$ ~

Solution:

Corresponds to smallest $x_B$, $x_A$, $\ldots$ $x_B$

Null

Fitting a Homography

- Estimating the transformation

Solution:

- Null-space vector of $A$

- Corresponds to smallest singular vector

$Ah = 0$

Minimizes least square error

Fitting a Homography

- Estimating the transformation

Uses: Analyzing Patterns and Shapes

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

Homography

The floor (enlarged)
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

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 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.
RANSAC Line Fitting Example
- Task: Estimate the best line

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.

"11 inlier points"

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: $(1 - w^n)^k$
  - Choose $k$ high enough to keep this below desired failure rate.

RANSAC: Computed $k$ (p=0.99)

<table>
<thead>
<tr>
<th>Sample size $n$</th>
<th>Proportion of outliers</th>
</tr>
</thead>
<tbody>
<tr>
<td>5% 10% 20% 25% 30% 40% 50%</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2 3 5 6 7</td>
</tr>
<tr>
<td>3</td>
<td>3 4 7 9 11</td>
</tr>
<tr>
<td>4</td>
<td>3 5 9 13 17</td>
</tr>
<tr>
<td>5</td>
<td>4 6 12 17 26</td>
</tr>
<tr>
<td>6</td>
<td>4 7 16 24 37</td>
</tr>
<tr>
<td>7</td>
<td>4 8 20 33 54</td>
</tr>
<tr>
<td>8</td>
<td>5 9 26 44 78</td>
</tr>
</tbody>
</table>

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

**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).
  - Of course, a hypothesis from a single match is unreliable.
  - Solution: let each match vote for its hypothesis in a Hough space with very coarse bins.

**Pose Clustering and Verification with SIFT**

- To detect instances of objects from a model base:
  1. Index descriptors
     - Distinctive features narrow down possible matches
### 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

### Location Recognition

- Training

### 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.

### 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
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).

  [Image 220x667 to 269x680]

- More details on RANSAC can also be found in Chapter 4.7 of Hartley & Zisserman.

  [Image 223x555 to 266x616]

  - 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