A Script...

- We’ve created a script... for the part of the lecture on object recognition & categorization
  - K. Grauman, B. Leibe
  - Visual Object Recognition

- Chapter 3: Local Feature Extraction (Last+this lecture)
- Chapter 4: Matching (Monday’s topic)
- Chapter 5: Geometric Verification (Wednesday’s topic)

- Available on the L2P -

Recap: Requirements for Local Features

- Problem 1: Detect the same point independently in both images
- Problem 2: For each point correctly recognize the corresponding one

We need a repeatable detector!
We need a reliable and distinctive descriptor!

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 Detector [Harris88]

1. Compute second moment matrix (autocorrelation matrix)
   \[ M(\sigma_x, \sigma_y) = g(\sigma_x) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} g(\sigma_y) \]
2. Square of derivatives
3. Gaussian filter \( g(\sigma) \)
4. Cornerness function - two strong eigenvalues
   \[ R = \text{det}(M(\sigma_x, \sigma_y)) - \alpha \text{trace}(M(\sigma_x, \sigma_y))^2 \]
   \[ = g(I_x^2) g(I_y^2) - [g(I_x I_y)]^2 - \alpha [g(I_x^2) + g(I_y^2)]^2 \]
5. Perform non-maximum suppression
Recap: Harris Detector Responses [Harris88]

Effect: A very precise corner detector.

Hessian Detector [Beaudet78]

- Hessian determinant

\[
\text{Hessian}(I) = \begin{bmatrix}
I_{xx} & I_{xy} \\
I_{xy} & I_{yy}
\end{bmatrix}
\]

Note: these are 2nd derivatives!

Intuition: Search for strong derivatives in two orthogonal directions

Hessian Detector - Responses [Beaudet78]

Effect: Responses mainly on corners and strongly textured areas.

Topics of This Lecture

- Local Feature Extraction (cont’d)
  - Scale Invariant Region Selection
  - Orientation normalization
  - Affine Invariant Feature Extraction

- Local Descriptors
  - SIFT
  - Applications

- Recognition with Local Features
  - Matching local features
  - Finding consistent configurations
  - Alignment: linear transformations
  - Affine estimation
  - Homography estimation
From Points to Regions...

- The Harris and Hessian operators define interest points.
  - Precise localization
  - High repeatability

- In order to compare those points, we need to compute a descriptor over a region.
  - How can we define such a region in a scale invariant manner?
  - I.e. how can we detect scale invariant interest regions?

Naïve Approach: Exhaustive Search

- Multi-scale procedure
  - Compare descriptors while varying the patch size

Naïve Approach: Exhaustive Search

- Multi-scale procedure
  - Compare descriptors while varying the patch size

<table>
<thead>
<tr>
<th>Similarity measure</th>
<th>$d(f_A, f_B)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_A$</td>
<td>$f_B$</td>
</tr>
</tbody>
</table>

Naïve Approach: Exhaustive Search

- Comparing descriptors while varying the patch size
  - Computationally inefficient
  - Inefficient but possible for matching
  - Prohibitive for retrieval in large databases
  - Prohibitive for recognition
Automatic Scale Selection

- **Solution:**
  - Design a function on the region, which is “scale invariant” (the same for corresponding regions, even if they are at different scales)
  - Example: average intensity. For corresponding regions (even of different sizes) it will be the same.
  - For a point in one image, we can consider it as a function of region size (patch width)

  \[ f \]

  [Image of graph showing function responses for increasing scale (scale signature)]

  \[ \text{Region size} \]

  \[ \text{Image 1} \]

  \[ \text{Region size} \]

  \[ \text{Image 2} \]

- **Function responses for increasing scale (scale signature)**

  [Image of graphs showing function responses for increasing scale (scale signature)]

  \[ f \]

  [Image of graph showing function responses for increasing scale (scale signature)]

  \[ \text{Region size} \]

  \[ \text{Image 1} \]

  \[ \text{Region size} \]

  \[ \text{Image 2} \]

  \[ f \]

  [Image of graph showing function responses for increasing scale (scale signature)]

  \[ \text{Region size} \]

  \[ \text{Image 1} \]

  \[ \text{Region size} \]

  \[ \text{Image 2} \]

  \[ f \]

  [Image of graph showing function responses for increasing scale (scale signature)]

  \[ \text{Region size} \]

  \[ \text{Image 1} \]

  \[ \text{Region size} \]

  \[ \text{Image 2} \]

  \[ f \]

  [Image of graph showing function responses for increasing scale (scale signature)]

  \[ \text{Region size} \]

  \[ \text{Image 1} \]

  \[ \text{Region size} \]

  \[ \text{Image 2} \]
Automatic Scale Selection

- Function responses for increasing scale (scale signature)

What Is A Useful Signature Function?

- Laplacian-of-Gaussian = “blob” detector

Characteristic Scale

- We define the characteristic scale as the scale that produces peak of Laplacian response

Laplacian-of-Gaussian (LoG)

- Interest points:
  - Local maxima in scale space of Laplacian-of-Gaussian

Laplacian-of-Gaussian (LoG)

- Interest points:
  - Local maxima in scale space of Laplacian-of-Gaussian

\[ L_\sigma(x, y) = \nabla^2 \left( \frac{1}{\sqrt{2\pi \sigma^2}} e^{-\frac{x^2+y^2}{2\sigma^2}} \right) \]

\[ L_\sigma(x, y) + L_{\sigma/\sqrt{2}}(x, y) \]

\[ \Rightarrow \text{List of } (x, y, \sigma) \]

LoG Detector: Workflow

Slide adapted from Krystian Mikolajczyk

B. Leibe

Slide credit: Svetlana Lazebnik
**Difference-of-Gaussian (DoG)**

- We can efficiently approximate the Laplacian with a difference of Gaussians:
  \[
  L = \sigma^2 \left( G_x(x, y, \sigma) + G_y(x, y, \sigma) \right) \quad \text{(Laplacian)}
  \]
  \[
  \text{DoG} = G(x, y, k\sigma) - G(x, y, \sigma) \quad \text{(Difference of Gaussians)}
  \]

- Advantages?
  - No need to compute 2nd derivatives.
  - Gaussians are computed anyway, e.g. in a Gaussian pyramid.

**Key point localization with DoG**

- Detect maxima of difference-of-Gaussian (DoG) in scale space
- Then reject points with low contrast (threshold)
- Eliminate edge responses

**DoG - Efficient Computation**

- Computation in Gaussian scale pyramid

**Results: Lowe’s DoG**

**Harris-Laplace** [Mikolajczyk ‘01]

1. Initialization: Multiscale Harris corner detection
2. Scale selection based on Laplacian
   (same procedure with Hessian \(\Rightarrow\) Hessian-Laplace)
Summary: Scale Invariant Detection

- **Given:** Two images of the same scene with a large scale difference between them.
- **Goal:** Find the same interest points independently in each image.
- **Solution:** Search for maxima of suitable functions in scale and in space (over the image).

Two strategies:
- Laplacian-of-Gaussian (LoG)
- Difference-of-Gaussian (DoG) as a fast approximation

These can be used either on their own, or in combinations with single-scale keypoint detectors (Harris, Hessian).

Topics of This Lecture

- **Local Feature Extraction (cont’d)**
  - Scale Invariant Region Selection
  - Orientation normalization
  - Affine Invariant Feature Extraction
- **Local Descriptors**
  - SIFT
  - Applications
- **Recognition with Local Features**
  - Matching local features
  - Finding consistent configurations
  - Alignment: linear transformations
  - Affine estimation
  - Homography estimation

Rotation Invariant Descriptors

- Find local orientation
  - Dominant direction of gradient for the image patch
- Rotate patch according to this angle
  - This puts the patches into a canonical orientation.

Orientation Normalization: Computation

- Compute orientation histogram
- Select dominant orientation
- Normalize: rotate to fixed orientation

The Need for Invariance

- Up to now, we had invariance to
  - Translation
  - Scale
  - Rotation
- Not sufficient to match regions under viewpoint changes
  - For this, we need also affine adaptation

Topics of This Lecture

- **Local Feature Extraction (cont’d)**
  - Scale Invariant Region Selection
  - Orientation normalization
  - Affine Invariant Feature Extraction
- **Local Descriptors**
  - SIFT
  - Applications
- **Recognition with Local Features**
  - Matching local features
  - Finding consistent configurations
  - Alignment: linear transformations
  - Affine estimation
  - Homography estimation
Affine Adaptation

- **Problem:**
  - Determine the characteristic shape of the region.
  - Assumption: shape can be described by “local affine frame”.
- **Solution: iterative approach**
  - Use a circular window to compute second moment matrix.
  - Compute eigenvectors to adapt the circle to an ellipse.
  - Recompute second moment matrix using new window and iterate...

Iterative Affine Adaptation

1. Detect keypoints, e.g. multi-scale Harris
2. Automatically select the scales
3. Adapt affine shape based on second order moment matrix
4. Refine point location

Affine Normalization/Deskewing

- **Steps**
  - Rotate the ellipse’s main axis to horizontal
  - Scale the x axis, such that it forms a circle

Affine Adaptation Example

- Scale-invariant regions (blobs)

Summary: Affine-Inv. Feature Extraction

- Extract affine regions
- Normalize regions
- Eliminate rotational ambiguity
- Compare descriptors
Invariance vs. Covariance

- **Invariance:**
  - features(transform(image)) = features(image)

- **Covariance:**
  - features(transform(image)) = transform(features(image))

Covariant detection ⇒ invariant description

Topics of This Lecture

- **Local Feature Extraction (cont’d)**
  - Orientation normalization
  - Affine Invariant Feature Extraction

- **Local Descriptors**
  - SIFT
  - Applications

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

Local Descriptors

- We know how to detect points
- Next question: How to describe them for matching?

Point descriptor should be:
1. Invariant
2. Distinctive

Feature Descriptors

- Disadvantage of patches as descriptors:
  1. Small shifts can affect matching score a lot

- **Solution:** histograms

Feature Descriptors: SIFT

- Scale Invariant Feature Transform
- Descriptor computation:
  1. Divide patch into 4x4 sub-patches: 16 cells
  2. Compute histogram of gradient orientations (8 reference angles) for all pixels inside each sub-patch
  3. Resulting descriptor: 4x4x8 = 128 dimensions

Overview: SIFT

- Extraordinarily robust matching technique
  - Can handle changes in viewpoint up to ~60 deg. out-of-plane rotation
  - Can handle significant changes in illumination
  - Sometimes even day vs. night (below)
  - Fast and efficient—can run in real time
  - Lots of code available

Working with SIFT Descriptors

- One image yields:
  - $n$ 2D points giving positions of the patches
    - $[n \times 2]$ matrix
  - $n$ scale parameters specifying the size of each patch
    - $[n \times 1]$ vector
  - $n$ orientation parameters specifying the angle of the patch
    - $[n \times 1]$ vector
  - $n$ 128-dimensional descriptors: each one is a histogram of the gradient orientations within a patch
    - $[n \times 128]$ matrix

Local Descriptors: SURF

- Fast approximation of SIFT idea
  - Efficient computation by 2D box filters & integral images
  - 6 times faster than SIFT
  - Equivalent quality for object identification
    - http://www.vision.ee.ethz.ch/~surf
  - GPU implementation available
    - Feature extraction @ 100Hz (detector + descriptor, 640×480 img)

You Can Try It At Home...

- For most local feature detectors, executables are available online:
  - http://robots.ox.ac.uk/~vgg/research/affine
  - http://www.cs.ubc.ca/~lowe/keypoints/
  - http://www.vision.ee.ethz.ch/~surf

Topics of This Lecture

- Local Feature Extraction (cont’d)
  - Orientation normalization
  - Affine Invariant Feature Extraction
- Local Descriptors
  - SIFT
  - Applications
- Recognition with Local Features
  - Matching local features
  - Finding consistent configurations
  - Alignment: linear transformations
  - Affine estimation
  - Homography estimation
Applications of Local Invariant Features

- Wide baseline stereo
- Motion tracking
- Panoramas
- Mobile robot navigation
- 3D reconstruction
- Recognition
  - Specific objects
  - Textures
  - Categories
- ...
Value of Local Features

- Advantages
  - Critical to find distinctive and repeatable local regions for multi-view matching.
  - Complexity reduction via selection of distinctive points.
  - Describe images, objects, parts without requiring segmentation; robustness to clutter & occlusion.
  - Robustness: similar descriptors in spite of moderate view changes, noise, blur, etc.

- How can we use local features for such applications?
  - Next: matching and recognition

Topics of This Lecture

- Local Feature Extraction (cont’d)
  - Orientation normalization
  - Affine Invariant Feature Extraction

- Local Descriptors
  - SIFT
  - Applications

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

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

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:
  \[ p' = Mp \]

What Can Be Represented by a 2x2 Matrix?

- 2D Scaling?
  \[ x' = s_x \cdot x \]
  \[ y' = s_y \cdot y \]

- 2D Rotation around (0,0)?
  \[ x' = \cos \theta \cdot x - \sin \theta \cdot y \]
  \[ y' = \sin \theta \cdot x + \cos \theta \cdot y \]

- 2D Shearing?
  \[ x' = x + s_h \cdot y \]
  \[ y' = y + s_v \cdot x \]
**What Can be Represented by a 2x2 Matrix?**

- **2D Mirror about y axis?**
  \[ x' = -x \quad y' = y \]
  \[ \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \]

- **2D Mirror over (0,0)?**
  \[ x' = -x \quad y' = -y \]
  \[ \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \]

- **2D Translation?**
  \[ x' = x + t_x \quad y' = y + t_y \]
  \[ \text{NO!} \]

**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:**
  \[
  \text{Translation} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}
  \]

**2D Linear Transforms**

- **Basic 2D transformations as 3x3 matrices**
  \[
  \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & 0 \\ c & d & 0 \\ e & f & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}
  \]
  - Only linear 2D transformations can be represented with a 2x2 matrix.
  - Linear transformations are combinations of...
    - Scale,
    - Rotation,
    - Shear, and
    - Mirror

**2D Affine Transformations**

- **Affine transformations** are combinations of...
  - Linear transformations, and
  - Translations
- Parallel lines remain parallel

**Projective Transformations**

- **Projective transformations:**
  - Affine transformations, and
  - Projective warps
- Parallel lines do not necessarily remain parallel
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").

**Alignment Problem**

- 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

Let's Start with Affine Transformations

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

Fitting an Affine Transformation

- Affine model approximates perspective projection of planar objects

Recall: Least Squares Estimation

- Set of data points: \((X_1, X'_1), (X_2, X'_2), \ldots\)
- Goal: a linear function to predict \(X\)'s from \(X\)s:

\[
Xa + b = X
\]

- We want to find \(a\) and \(b\).
- How many \((X, X')\) pairs do we need?

\[
\begin{bmatrix} X_1 & 1 \\ X_2 & 1 \\ \vdots & \vdots \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} X'_1 \\ X'_2 \\ \vdots \end{bmatrix}, \quad Ax = B
\]

- What if the data is noisy?

\[
\begin{bmatrix} X_1 & 1 \\ X_2 & 1 \\ \vdots & \vdots \end{bmatrix} \begin{bmatrix} X'_1 \\ X'_2 \\ \vdots \end{bmatrix} = \min ||Ax - B||^2
\]

**Matlab:**

\[
x = A \backslash B
\]

\[
\begin{bmatrix} x'_1 \\ y'_1 \\ \vdots \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \\ \vdots & \vdots \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}
\]

Fitting an Affine Transformation

- assuming we know the correspondences, how do we get the 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 \((x_{new}, y_{new})\)?

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 preserved, but must preserve straight lines
- This is called a homography

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

Set scale factor to 1 ⇒ 8 parameters left.

Fitting a Homography

- Estimating the transformation

\[
\begin{bmatrix}
x' \\
y' \\
1
\end{bmatrix} =
\begin{bmatrix}
A & 0 & x_0 \\
0 & A & y_0 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
1
\end{bmatrix}
\]

Matrix notation
\[
x' = Hx
\]

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

Solution:

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

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

\[ x' = B'Ax \]

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

Corresponds to smallest image plane below

Fitting a Homography

Matrix notation

\[ x' = Hx \]

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

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

Homogenous coordinates

\[ x_h = \begin{bmatrix} x \\ h_x \\ h_y \end{bmatrix} \]

\[ y_h = \begin{bmatrix} y \\ h_x \\ h_y \end{bmatrix} \]

Image coordinates

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

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

Fitting a Homography

Null-space vector of A

Solution:

Corresponds to smallest eigenvector

\[ A = UDV^T \]

\[ h = \begin{bmatrix} \ldots \\ \ldots \\ \ldots \\ \ldots \\ \ldots \\ \ldots \end{bmatrix} \]

Minimizes least square error

Image Warping with Homographies
Uses: Analyzing Patterns and Shapes

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

Analyzing Patterns and Shapes

From Martin Kemp. The Science of Art (manual reconstruction)

Summary: Recognition by Alignment

- Basic matching algorithm
  1. Detect interest points in two images.
  2. Extract patches and compute a descriptor for each one.
  3. Compare one feature from image 1 to every feature in image 2 and select the nearest-neighbor pair.
  4. Repeat the above for each feature from image 1.
  5. Use the list of best pairs to estimate the transformation between images.

- Transformation estimation
  - Affine
  - Homography

Time for a Demo...

Automatic panorama stitching

References and Further Reading

- More details on homography estimation can be found in Chapter 4.7 of
  - R. Hartley, A. Zisserman. Multiple View Geometry in Computer Vision. 2nd Ed., Cambridge Univ. Press, 2004

- Details about the DoG detector and the SIFT descriptor can be found in
  - D. Lowe. Distinctive image features from scale-invariant keypoints. IJCV 60(2), pp. 91-110, 2004

- Try the available local feature detectors and descriptors
  - http://www.robots.ox.ac.uk/~vgg/research/affine/detectors.html#binaries