Computer Vision - Lecture 12
Object Recognition with Local Features

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

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
• Object Categorization II
  • Part based Approaches
• 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: Requirements for Local Features
• Problem 1: Detect the same point independently in both images
• Problem 2: For each point correctly recognize the corresponding one

Recap: Harris Detector
• Compute second moment matrix (autocorrelation matrix)

Recap: Harris Detector Responses
Recap: Hessian Detector [Beaudet78]

- Hessian determinant

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

\[ \det(\text{Hessian}(I)) = I_{xx}I_{yy} - I_{xy}^2 \]

In Matlab:

\[ I_{xx} I_{yy} - (I_{xy})^2 \]

Recap: Hessian Detector Responses [Beaudet78]

Effect: Responses mainly on corners and strongly textured areas.

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

\[ L = \sigma^2 (G_x(x, y, \sigma) + G_y(x, y, \sigma)) \]  
\[ \text{(Laplacian)} \]

\[ \text{DoG} = G(x, y, k\sigma) - G(x, y, \sigma) \]  
\[ \text{(Difference of Gaussians)} \]

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

Recap: LoG Detector Responses

Difference-of-Gaussian (DoG)

- We can efficiently approximate the Laplacian with a difference of Gaussians:

\[ L = \sigma^2 (G_x(x, y, \sigma) + G_y(x, y, \sigma)) \]  
\[ \text{(Laplacian)} \]

\[ \text{DoG} = G(x, y, k\sigma) - G(x, y, \sigma) \]  
\[ \text{(Difference of Gaussians)} \]
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
  - Similar criterion as with Harris (eigenvalues of autocorrelation matrix)

Candidate keypoints: list of \((x, y, \sigma)\)

DoG - Efficient Computation

- Computation in Gaussian scale pyramid

Results: Lowe’s DoG

Combination: Harris-Laplace [Mikolajczyk ‘01]

1. Initialization: Multiscale Harris corner detection
2. Scale selection based on Laplacian
   - Take points that are simultaneous maxima of Harris and LoG.
     (same procedure with Hessian ⇒ 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)
  - 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 [Lowe, SIFT, 1999]
- 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

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))

**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?
Local Descriptors
- Simplest descriptor: list of intensities within a patch.
- What is this going to be invariant to?

Feature Descriptors
- Disadvantage of patches as descriptors:
  - Small shifts can affect matching score a lot

Feature Descriptors: SIFT
- 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

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

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, 640x480 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)
  - Creation normalization
  - Affine Invariant Feature Extraction

- Local Descriptors
  - SIFT
  - Applications
  - Recognition with Local Features
    - Matching local features
    - Finding consistent configurations
    - Alignments: 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

Wide-Baseline Stereo

Automatic Mosaicing

http://www.robots.ox.ac.uk/~vgg/research/affine/detectors.html#binaries
Panorama Stitching

http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html

Recognition of Specific Objects, Scenes

Schmid and Mohr 1997

Sivic and Zisserman, 2003

Rothganger et al. 2003

Lowe 2002

Recognition of Categories

Constellation model

Bags of words

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?
  - This week: 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
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

\[ p = (x, y) \]
\[ \rightarrow T \]
\[ p' = (x', y') \]

- 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:
  \[ p' = M p, \]
  \[ \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \]

What Can be Represented by a 2x2 Matrix?

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

- 2D Rotation around (0,0)?
  \[ x' = \cos \theta x - \sin \theta y \]
  \[ y' = \sin \theta x + \cos \theta y \]
  \[ \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \]

- 2D Shearing?
  \[ x' = x + sh_y y \]
  \[ y' = sh_x x + y \]
  \[ \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & sh_y \\ sh_x & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \]

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

What Can be Represented by a 2x2 Matrix?

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

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

- 2D Translation?
  \[ x' = x + t_x \]
  \[ y' = y + t_y \]
  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:
  \[ \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \]

Slide credit: Kristen Grauman

Slide credit: Alexej Efros

Slide credit: Alexej Efros
Basic 2D Transformations
- Basic 2D transformations as 3x3 matrices

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

- Translation
- Rotation
- Shearing

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

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

- Parallel lines remain parallel

2D Affine Transformations
- Affine transformations:
  - Linear transformations, and
  - Translations

- Parallel lines 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?

\[
\begin{bmatrix}
    x'_i \\
    y'_i
\end{bmatrix} =
\begin{bmatrix}
    m_1 & m_2 & m_3 & 1 \\
    m_4 & m_5 & m_6 & 1
\end{bmatrix}
\begin{bmatrix}
    x_i \\
    y_i
\end{bmatrix} +
\begin{bmatrix}
    t_1 \\
    t_2
\end{bmatrix}
\]

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

Recall: Least Squares Estimation

- Set of data points: \((X_1, X_2), (X_3, X_4)\)
- 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?
  \[ X_i a + b = X'_i \]
  \[ X_j a + b = X'_j \]
- What if the data is noisy?

\[
\begin{bmatrix}
    X_i & 1 & X'_i \\
    X_j & 1 & X'_j
\end{bmatrix}
\begin{bmatrix}
    a \\
    b
\end{bmatrix}
=\begin{bmatrix}
    X'_i \\
    X'_j
\end{bmatrix}
\]

\[ Ax = B \]

Matlab:
\[ x = A \backslash B \]

\[ \Rightarrow \text{Least-squares minimization} \]

Set scale factor to 1 ⇒ 8 parameters left.
Fitting a Homography

• Estimating the transformation

Homogenous coordinates

Image coordinates

Matrix notation

$\mathbf{X} \leftrightarrow \mathbf{X}_h$

$\mathbf{X} = \mathbf{A} \mathbf{X}_h$

$x' = Hx$

$x^* = \frac{1}{\lambda} x'$

Fitting a Homography

• Estimating the transformation

Homogenous coordinates

Image coordinates

Matrix notation

$\mathbf{X} \leftrightarrow \mathbf{X}_h$

$\mathbf{X} = \mathbf{A} \mathbf{X}_h$

$x' = Hx$

$x^* = \frac{1}{\lambda} x'$

Fitting a Homography

• Estimating the transformation

Homogenous coordinates

Image coordinates

Matrix notation

$\mathbf{X} \leftrightarrow \mathbf{X}_h$

$\mathbf{X} = \mathbf{A} \mathbf{X}_h$

$x' = Hx$

$x^* = \frac{1}{\lambda} x'$

Fitting a Homography

• Estimating the transformation

Homogenous coordinates

Image coordinates

Matrix notation

$\mathbf{X} \leftrightarrow \mathbf{X}_h$

$\mathbf{X} = \mathbf{A} \mathbf{X}_h$

$x' = Hx$

$x^* = \frac{1}{\lambda} x'$
Fitting a Homography

- Estimating the transformation

\[ A h = 0 \]

Solution:
- Null-space vector of A
- Corresponds to smallest eigenvector

Uses: Analyzing Patterns and Shapes

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

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