Computer Vision - Lecture 12

Local Features II

10.12.2015

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de
leibe@vision.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
  - Deep Learning Approaches
• 3D Reconstruction
• Motion and Tracking
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+this lecture)
- Chapter 4: Matching (Tuesday’s topic)
- Chapter 5: Geometric Verification (Thursday’s topic)

- Available on the L2P -
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

\[ d(f_A, f_B) < T \]
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: Harris Detector [Harris88]

- Compute second moment matrix (autocorrelation matrix)

\[ M(\sigma_I, \sigma_D) = g(\sigma_I) \ast \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix} \]

1. Image derivatives
2. Square of derivatives
3. Gaussian filter \( g(\sigma_I) \)
4. Cornerness function - two strong eigenvalues

\[ R = \det[M(\sigma_I, \sigma_D)] - \alpha[\text{trace}(M(\sigma_I, \sigma_D))]^2 \]
\[ = g(I_x^2)g(I_y^2) - [g(I_x I_y)]^2 - \alpha[g(I_x^2) + g(I_y^2)]^2 \]

5. Perform non-maximum suppression

Slide credit: Krystian Mikolajczyk
Recap: Harris Detector Responses [Harris88]

Effect: A very precise corner detector.

Slide credit: Krystian Mikolajczyk
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:

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

Slide credit: Krystian Mikolajczyk
Recap: Hessian Detector Responses [Beaudet78]

Effect: Responses mainly on corners and strongly textured areas.

Slide credit: Krystian Mikolajczyk
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
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

\[ L_{xx}(\sigma) + L_{yy}(\sigma) \]

\[ \Rightarrow \text{List of } (x, y, \sigma) \]
Recap: LoG Detector Responses
Difference-of-Gaussian (DoG)

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

\[ L = \sigma^2 \left( G_{xx}(x, y, \sigma) + G_{yy}(x, y, \sigma) \right) \]

(Laplacian)

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

(Difference of Gaussians)

- Advantages?
  - No need to compute 2\textsuperscript{nd} 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

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

Slide credit: David Lowe
DoG - Efficient Computation

- Computation in Gaussian scale pyramid

Slide adapted from Krystian Mikolajczyk
Results: Lowe’s DoG
Harris-Laplace [Mikolajczyk ‘01]

1. Initialization: Multiscale Harris corner detection

Slide adapted from Krystian Mikolajczyk
Harris-Laplace [Mikolajczyk ‘01]

1. Initialization: Multiscale Harris corner detection
2. Scale selection based on Laplacian
   (same procedure with Hessian ⇒ Hessian-Laplace)

Harris points

Harris-Laplace points

Slide adapted from Krystian Mikolajczyk

B. Leibe
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
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
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…

Slide adapted from Svetlana Lazebnik
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


Slide credit: Tinne Tuytelaars
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)

Slide credit: Svetlana Lazebnik
Affine Adaptation Example

Affine-adapted blobs

Slide credit: Svetlana Lazebnik
Summary: Affine-Inv. Feature Extraction

Extract affine regions → Normalize regions → Eliminate rotational ambiguity → Compare descriptors

Slide credit: Svetlana Lazebnik

B. Leibe
Invariance vs. Covariance

- **Invariance:**
  - \( \text{features}(\text{transform}(\text{image})) = \text{features}(\text{image}) \)

- **Covariance:**
  - \( \text{features}(\text{transform}(\text{image})) = \text{transform}(\text{features}(\text{image})) \)

**Covariant detection \( \Rightarrow \) invariant description**

Slide credit: Svetlana Lazebnik, David Lowe
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
Local Descriptors

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

Write regions as vectors

\[ A \rightarrow a, \ B \rightarrow b \]
Feature Descriptors

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

- Solution: histograms
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


Slide credit: Svetlana Lazebnik
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
    - $[n \times 2 \text{ matrix}]$
  - $n$ scale parameters specifying the size of each patch
    - $[n \times 1 \text{ vector}]$
  - $n$ orientation parameters specifying the angle of the patch
    - $[n \times 1 \text{ vector}]$
  - $n$ 128-dimensional descriptors: each one is a histogram of the gradient orientations within a patch
    - $[n \times 128 \text{ matrix}]$
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
    - [Bay, ECCV'06], [Cornelis, CVGPU’08]
  - [Bay, ECCV'06], [Cornelis, CVGPU’08]

- GPU implementation available
  - Feature extraction @ 100Hz
    (detector + descriptor, 640×480 img)
  - [Bay, ECCV'06], [Cornelis, CVGPU’08]
You Can Try It At Home...

• For most local feature detectors, executables are available online:
  • [http://robots.ox.ac.uk/~vgg/research/affine](http://robots.ox.ac.uk/~vgg/research/affine)
  • [http://www.vision.ee.ethz.ch/~surf](http://www.vision.ee.ethz.ch/~surf)
Affine Covariant Features

Detector output

<table>
<thead>
<tr>
<th>format:</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td></td>
</tr>
<tr>
<td>u_a v_a b_a c_a</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>u_m v_m a_m b_m c_m</td>
<td></td>
</tr>
</tbody>
</table>

Image with displayed regions

display_features.m

Parameters defining an affine region

\[(a-x-u)(x-u)+2b(x-u)(y-v) + c (y-v) (y-v) = 1\]

with \((0,0)\) at image top left corner

Code

- provided by the authors, see publications for details and links to authors web sites

Linux binaries

<table>
<thead>
<tr>
<th>Harris-Affine &amp; Herogan-Affine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example of use</td>
</tr>
<tr>
<td>prompt.x/h_affine.in -hareff -i img1.ppm -o img1.hareff -thres 1000</td>
</tr>
<tr>
<td>prompt.x/h_affine.in -hareff -i img1.ppm -o img1.hareff -thres 500</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MSER - Maximally stable extremal regions (also Windows)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example of use</td>
</tr>
<tr>
<td>prompt.x/mser.in -t 2 -es 2 -i img1.ppm -o img1.mser</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EBK - Intensity extrema based detector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example of use</td>
</tr>
<tr>
<td>prompt.x/ebk.in img1.ppm img1.ebk -scalefactor 1.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EBR - Edge based detector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example of use</td>
</tr>
<tr>
<td>prompt.x/ebk.in img1.ppm img1.ebr</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Salient region detector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example of use</td>
</tr>
<tr>
<td>prompt.x/salient.in img1.ppm img1.sal</td>
</tr>
</tbody>
</table>

Displaying 1

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

Slide credit: Kristen Grauman
Wide-Baseline Stereo
Automatic Mosaicing

B. Leibe

[Brown & Lowe, ICCV’03]
Panorama Stitching

(a) Matier data set (7 images)

(b) Matier final stitch

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

[Brown, Szeliski, and Winder, 2005]
Recognition of Specific Objects, Scenes

Schmid and Mohr 1997

Sivic and Zisserman, 2003

Rothganger et al. 2003

Lowe 2002

Slide credit: Kristen Grauman
Recognition of Categories

Constellation model

Bags of words

<table>
<thead>
<tr>
<th>Database</th>
<th>Sample cluster #1</th>
<th>Sample cluster #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Airplanes</td>
<td><img src="image" alt="Airplanes_cluster_1" /> <img src="image" alt="Airplanes_cluster_2" /></td>
<td><img src="image" alt="Airplanes_cluster_3" /> <img src="image" alt="Airplanes_cluster_4" /></td>
</tr>
<tr>
<td>Motorbikes</td>
<td><img src="image" alt="Motorbikes_cluster_1" /> <img src="image" alt="Motorbikes_cluster_2" /></td>
<td><img src="image" alt="Motorbikes_cluster_3" /> <img src="image" alt="Motorbikes_cluster_4" /></td>
</tr>
<tr>
<td>Leaves</td>
<td><img src="image" alt="Leaves_cluster_1" /> <img src="image" alt="Leaves_cluster_2" /></td>
<td><img src="image" alt="Leaves_cluster_3" /> <img src="image" alt="Leaves_cluster_4" /></td>
</tr>
<tr>
<td>Wild Cats</td>
<td><img src="image" alt="WildCats_cluster_1" /> <img src="image" alt="WildCats_cluster_2" /></td>
<td><img src="image" alt="WildCats_cluster_3" /> <img src="image" alt="WildCats_cluster_4" /></td>
</tr>
<tr>
<td>Faces</td>
<td><img src="image" alt="Faces_cluster_1" /> <img src="image" alt="Faces_cluster_2" /></td>
<td><img src="image" alt="Faces_cluster_3" /> <img src="image" alt="Faces_cluster_4" /></td>
</tr>
<tr>
<td>Bicycles</td>
<td><img src="image" alt="Bicycles_cluster_1" /> <img src="image" alt="Bicycles_cluster_2" /></td>
<td><img src="image" alt="Bicycles_cluster_3" /> <img src="image" alt="Bicycles_cluster_4" /></td>
</tr>
<tr>
<td>People</td>
<td><img src="image" alt="People_cluster_1" /> <img src="image" alt="People_cluster_2" /></td>
<td><img src="image" alt="People_cluster_3" /> <img src="image" alt="People_cluster_4" /></td>
</tr>
</tbody>
</table>

Weber et al. (2000)
Fergus et al. (2003)

Csurka et al. (2004)
Dorko & Schmid (2005)
Sivic et al. (2005)
Lazebnik et al. (2006), ...

Slide credit: Svetlana Lazebnik
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

Local Features, e.g. SIFT

Slide credit: David Lowe
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?

Slide credit: Kristen Graum
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:

$$p' = Mp \ , \ \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} M$$

Slide credit: Alexej Efros
What Can be Represented by a $2 \times 2$ Matrix?

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

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

- **2D Shearing?**
  \[
  x' = x + s_{h_x} \times y \\
  y' = s_{h_y} \times x + y
  \]
  \[
  \begin{bmatrix}
  x' \\
  y'
  \end{bmatrix} =
  \begin{bmatrix}
  1 & s_{h_x} \\
  s_{h_y} & 1
  \end{bmatrix}
  \begin{bmatrix}
  x \\
  y
  \end{bmatrix}
  \]
What Can be Represented by a $2 \times 2$ 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!**
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?
  \[ x' = x + t_x \]
  \[ y' = y + t_y \]

- A: Using the rightmost column:
  \[
  \begin{bmatrix}
  1 & 0 & t_x \\
  0 & 1 & t_y \\
  0 & 0 & 1 \\
  \end{bmatrix}
  \]

Slide credit: Alexej Efros
Basic 2D Transformations

• Basic 2D transformations as 3x3 matrices

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

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

Rotation

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

Shearing

\[
\begin{bmatrix}
    x' \\
    y' \\
1
\end{bmatrix} =
\begin{bmatrix}
    1 & sh_x & 0 \\
    sh_y & 1 & 0 \\
    0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
    x \\
    y \\
1
\end{bmatrix}
\]
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”).

\[ x_i \quad \rightarrow \quad x_i' \quad T \]
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?

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

- Set of data points: \((X_1, X'_1), (X_2, X'_2), (X_3, X'_3)\)
- Goal: a linear function to predict \(X\)'s from \(Xs\):
  \[ Xa + b = X' \]
- We want to find \(a\) and \(b\).
- How many \((X, X')\) pairs do we need?
  \[
  \begin{align*}
  X_1a + b &= X'_1 \\
  X_2a + b &= X'_2 \\
  \end{align*}
  \]
  \[
  \begin{bmatrix}
  X_1 & 1 \\
  X_2 & 1 \\
  X_3 & 1 \\
  \vdots & \vdots
  \end{bmatrix}
  \begin{bmatrix}
  a \\
  b
  \end{bmatrix}
  =
  \begin{bmatrix}
  X'_1 \\
  X'_2 \\
  X'_3 \\
  \vdots
  \end{bmatrix}
  \]
  \[ Ax = B \]
- What if the data is noisy?
  \[
  \min \| Ax - B \|^2
  \]
  \[ \Rightarrow \text{Least-squares minimization} \]

Matlab:
\[ x = A \backslash B \]
Fitting an Affine Transformation

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

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

B. Leibe
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 $(x_{new}, y_{new})$?
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

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

Slide adapted from Alexej Efros
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

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

- Set scale factor to 1 ⇒ 8 parameters left.

Slide adapted from Alexej Efros
Fitting a Homography

- Estimating the transformation

Matrix notation

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

Homogenous coordinates

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

Image coordinates

\[
\begin{bmatrix}
    x' = Hx \\
y' \\
z'
\end{bmatrix}
\]

Slide credit: Krystian Mikolajczyk
Fitting a Homography

- Estimating the transformation

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

Image coordinates

\[
\begin{pmatrix}
x'' \\
y''
\end{pmatrix} = \frac{1}{z'}
\begin{pmatrix}
x' \\
y'
\end{pmatrix}
\]

Matrix notation

\[
x' = Hx
\]

\[
x'' = \frac{1}{z'} x'
\]
Fitting a Homography

- Estimating the transformation

Homogenous coordinates

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

Image coordinates

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

Matrix notation

\[
x' = Hx
\]

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

Slide credit: Krystian Mikolajczyk
Fitting a Homography

- Estimating the transformation

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

\[
x_A' = \frac{h_{11} x_B + h_{12} y_B + h_{13}}{h_{31} x_B + h_{32} y_B + 1}
\]

\[
y_A' = \frac{h_{21} x_B + h_{22} y_B + h_{23}}{h_{31} x_B + h_{32} y_B + 1}
\]

Image coordinates

Homogenous coordinates

Matrix notation

\[
x' = Hx
\]

\[
x'' = \frac{1}{z'} x'
\]
Fitting a Homography

- Estimating the transformation

\[ x_{A_i} = \frac{h_{11} x_{B_i} + h_{12} y_{B_i} + h_{13}}{h_{31} x_{B_i} + h_{32} y_{B_i} + 1} \]

\[ y_{A_i} = \frac{h_{21} x_{B_i} + h_{22} y_{B_i} + h_{23}}{h_{31} x_{B_i} + h_{32} y_{B_i} + 1} \]

Slide credit: Krystian Mikolajczyk
Fitting a Homography

- Estimating the transformation

\[ x_{A_i} = \frac{h_{11} x_{B_i} + h_{12} y_{B_i} + h_{13}}{h_{31} x_{B_i} + h_{32} y_{B_i} + 1} \]

\[ y_{A_i} = \frac{h_{21} x_{B_i} + h_{22} y_{B_i} + h_{23}}{h_{31} x_{B_i} + h_{32} y_{B_i} + 1} \]

Image coordinates

Homogenous coordinates

\[ x_{A_i} h_{31} x_{B_i} + x_{A_i} h_{32} y_{B_i} + x_{A_i} = h_{11} x_{B_i} + h_{12} y_{B_i} + h_{13} \]

\[ h_{11} x_{B_i} + h_{12} y_{B_i} + h_{13} - x_{A_i} h_{31} x_{B_i} - x_{A_i} h_{32} y_{B_i} - x_{A_i} = 0 \]

\[ h_{21} x_{B_i} + h_{22} y_{B_i} + h_{23} - y_{A_i} h_{31} x_{B_i} - y_{A_i} h_{32} y_{B_i} - y_{A_i} = 0 \]
Fitting a Homography

- Estimating the transformation

\[ \begin{align*}
  h_{11} x_{B_i} + h_{12} y_{B_i} + h_{13} - x_{A_i} h_{31} x_{B_i} - x_{A_i} h_{32} y_{B_i} - x_{A_i} &= 0 \\
  h_{21} x_{B_i} + h_{22} y_{B_i} + h_{23} - y_{A_i} h_{31} x_{B_i} - y_{A_i} h_{32} y_{B_i} - y_{A_i} &= 0
\end{align*} \]

\[ \begin{bmatrix}
  x_{B_1} & y_{B_1} & 1 & 0 & 0 & 0 & -x_{A_1} x_{B_1} & -x_{A_1} y_{B_1} & -x_{A_1} \\
  0 & 0 & 0 & x_{B_1} & y_{B_1} & 1 & -y_{A_1} x_{B_1} & -y_{A_1} y_{B_1} & -y_{A_1}
\end{bmatrix} \begin{bmatrix}
  h_{11} \\
  h_{12} \\
  h_{13} \\
  h_{21} \\
  h_{22} \\
  h_{23} \\
  h_{31} \\
  h_{32} \\
  1
\end{bmatrix} = \begin{bmatrix}
  0 \\
  0 \\
  . \\
  . \\
  . \\
  . \\
  . \\
  . \\
  1
\end{bmatrix} \]

\[ Ah = 0 \]
Fitting a Homography

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

\[
A = UDV^T = \begin{bmatrix}
  d_{11} & \cdots & d_{19} \\
  \vdots & \ddots & \vdots \\
  d_{91} & \cdots & d_{99}
\end{bmatrix}
\begin{bmatrix}
  v_{11} & \cdots & v_{19} \\
  \vdots & \ddots & \vdots \\
  v_{91} & \cdots & v_{99}
\end{bmatrix}^T
\]

\[h = \begin{bmatrix}
  v_{19}, \cdots, v_{99} \\
  v_{99}
\end{bmatrix}
\]

Minimizes least square error

Slide credit: Krystian Mikolajczyk
Image Warping with Homographies

Image plane in front

Black area where no pixel maps to

Slide credit: Steve Seitz
Uses: Analyzing Patterns and Shapes

• What is the shape of the b/w floor pattern?
Analyzing Patterns and Shapes

Automatic rectification

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