CAP 4453
Robot Vision
Dr. Gonzalo Vaca-Castaño
gonzalo.vacacastano@ucf.edu
Credits

• Some slides comes directly from these sources:
  • Ioannis (Yannis) Gkioulekas (CMU)
  • Kris Kitani.
  • Fredo Durand (MIT).
  • James Hays (Georgia Tech).
  • Yogesh S Rawat (UCF)
  • Noah Snavely (Cornell)
Short Review from last class
Image warping

How do we find point correspondences automatically?
Corner Detection: Basic Idea

• We should easily recognize the point by looking through a small window
• Shifting a window in *any direction* should give *a large change* in intensity

“flat” region: no change in all directions

“edge”: no change along the edge direction

“corner”: significant change in all directions

Source: A. Efros
Harris Detector

1. Compute x and y derivatives of image

\[ I_x = G_{\sigma}^x \ast I \quad I_y = G_{\sigma}^y \ast I \]

2. Compute products of derivatives at every pixel

\[ I_{x^2} = I_x \cdot I_x \quad I_{y^2} = I_y \cdot I_y \quad I_{xy} = I_x \cdot I_y \]

3. Compute the sums of the products of derivatives at each pixel

\[ S_{xy} = G_{\sigma}^x \ast I_{xy} \quad S_{y^2} = G_{\sigma}^y \ast I_{y^2} \quad S_{x^2} = G_{\sigma}^x \ast I_{x^2} \]

4. Define the matrix at each pixel

\[ M(x, y) = \begin{bmatrix} S_{x^2}(x, y) & S_{xy}(x, y) \\ S_{xy}(x, y) & S_{y^2}(x, y) \end{bmatrix} \]

5. Compute the response of the detector at each pixel

\[ R = \det M - k(\text{trace}M)^2 \]

6. Threshold on value of R; compute non-max suppression.
Use threshold on eigenvalues to detect corners

Think of a function to score ‘cornerness’
Harris & Stephens (1988)

\[ R = \text{det}(M) - \kappa \text{trace}^2(M) \]

Kanade & Tomasi (1994)

\[ R = \min(\lambda_1, \lambda_2) \]

Nobel (1998)

\[ R = \frac{\text{det}(M)}{\text{trace}(M) + \epsilon} \]

\[
\begin{align*}
\text{det } M &= \lambda_1 \lambda_2 \\
\text{trace } M &= \lambda_1 + \lambda_2 \\
\det \left( \begin{bmatrix} a & b \\ c & d \end{bmatrix} \right) &= ad - bc \\
\text{trace} \left( \begin{bmatrix} a & b \\ c & d \end{bmatrix} \right) &= a + d
\end{align*}
\]
Harris corner detection and translation

• What happens if image is translated?
• Derivatives, second moment matrix obtained through convolution, which is translation equivariant
• Eigenvalues based only on derivatives so cornerness is invariant
• Thus Harris corner detection location is equivariant to translation, and response is invariant to translation
What about rotation?

• Now every patch is rotated, so problem?

• Recall properties of second moment matrix

• Eigenvectors and eigenvalues of M
  • Define shift directions with the smallest and largest change in error
  • $x_{\text{max}}$ = direction of largest increase in E (across the edge)
  • $\lambda_{\text{max}}$ = amount of increase in direction $x_{\text{max}}$
  • $x_{\text{min}}$ = direction of smallest increase in E (along the edge)
  • $\lambda_{\text{min}}$ = amount of increase in direction $x_{\text{min}}$
What about rotation?

- What happens to eigenvalues and eigenvectors when a patch rotates?
- Eigenvectors represent the direction of maximum / minimum change in appearance, so they rotate with the patch.
- Eigenvalues represent the corresponding magnitude of maximum/minimum change so they stay constant.
- Corner response is only dependent on the eigenvalues so is invariant to rotation.
- Corner location is as before equivariant to rotation.
What about scaling?

- What was one patch earlier is now many

All points will be classified as edges

Not invariant to scaling
implementation

For each level of the Gaussian pyramid

compute feature response (e.g. Harris, Laplacian)

For each level of the Gaussian pyramid

if local maximum and cross-scale

save scale and location of feature \((x, y, s)\)
Implementation

- Instead of computing $f$ for larger and larger windows, we can implement using a fixed window size with a Gaussian pyramid
Blob detection
Laplacian of Gaussian

• “Blob” detector

• Find maxima \textit{and minima} of LoG operator in space and scale
Characteristic scale

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

optimal scale

2.1  4.2  6.0  9.8  15.5  17.0

Full size image

2.1  4.2  6.0  9.8  15.5  17.0

3/4 size image
cross-scale maximum

local maximum

local maximum

local maximum

4.2

6.0

9.8
Robot Vision

11. Feature points description
Outline

• Motivation
• Detecting key points
  • Harris corner detector
  • Blob detection
• Feature descriptors
  • HOG
  • MOPS
• SIFT
Matching feature points

We know how to detect good points
Next question: **How to match them?**

Two interrelated questions:
1. How do we *describe* each feature point?
2. How do we *match* descriptions?
Feature descriptor

$x_1$  $x_2$

$y_1$  $y_2$
Feature matching

\[
\begin{align*}
&\begin{array}{c|c}
  y_1 & y_2 \\
  \hline
  x_1 & d(x_1, y_1) & d(x_1, y_2) \\
  x_2 & d(x_2, y_1) & d(x_2, y_2)
\end{array}
\end{align*}
\]
Feature Descriptor

\[ x_1 \quad x_2 \]
Feature detection and description

• Harris corner detection gives:
  • Location of each detected corner
  • Orientation of the corner (given by $x_{\text{max}}$)
  • Scale of the corner (the image scale which gives the maximum response at this location)

• Want feature descriptor that is
  • Invariant to photometric transformations, translation, rotation, scaling
  • Discriminative
Multiscale Oriented PatcheS descriptor

• Describe a corner by the patch around that pixel
• Scale invariance by using scale identified by corner detector
• Rotation invariance by using orientation identified by corner detector
• Photometric invariance by subtracting mean and dividing by standard deviation
Multiscale Oriented PatcheS descriptor

- Take 40x40 square window around detected feature at the right scale
- Scale to 1/5 size (using prefiltering)
- Rotate to horizontal
- Sample 8x8 square window centered at feature
- Intensity normalize the window by subtracting the mean, dividing by the standard deviation in the window

Adapted from slide by Matthew Brown
Figure 1. Multi-scale Oriented Patches (MOPS) extracted at five pyramid levels from one of the Matier images. The boxes show the feature orientation and the region from which the descriptor vector is sampled.
Towards a better feature descriptor

• Match *pattern of edges*
  • Edge orientation – clue to shape
  • Invariant to almost all photometric transformations

• Be resilient to *small deformations*
  • Deformations might move pixels around, but slightly
  • Deformations might change edge orientations, but slightly
Invariance to deformation

• Precise edge orientations are not resilient to out-of-plane rotations and deformations
• But we can *quantize* edge orientation: only record rough orientation

Between 30 and 45
Invariance to deformation

\[ g(\theta) = \begin{cases} 
0 & \text{if } 0 < \theta < \frac{2\pi}{N} \\
1 & \text{if } \frac{2\pi}{N} < \theta < \frac{4\pi}{N} \\
2 & \text{if } \frac{4\pi}{N} < \theta < \frac{6\pi}{N} \\
N - 1 & \text{if } 2(N - 1)\pi/N \end{cases} \]
Invariance to deformation

- Deformation can also move pixels around
- Again, instead of precise location of each pixel, only want to record rough location
- Divide patch into a grid of cells
- Record counts of each orientation in each cell: orientation histograms
Histogram of Oriented Gradients (HOG)

- Revisiting histogram

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>0</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>

image

histogram
Histogram of Oriented Gradients (HOG)

- Given an image $I$, and a pixel location $(i,j)$.
- We want to compute the HOG feature for that pixel.
- The main operations can be described as a sequence of five steps.
Histogram of Oriented Gradients (HOG)

• Step 1: Extract a square window (called “block”) of some size.
Histogram of Oriented Gradients (HOG)

• Step 2: Divide block into a square grid of sub-blocks (called “cells”) (2x2 grid in our example, resulting in four cells).
Histogram of Oriented Gradients (HOG)

• Step 3: Compute orientation histogram of each cell.

$$\theta = \tan^{-1} \frac{f_x}{f_y}$$
Histogram of Oriented Gradients (HOG)

• Step 3: Compute orientation histogram of each cell.

\[ \theta = \tan^{-1} \left( \frac{f_x}{f_y} \right) \]

• Cell size is 8x8

• Quantize the gradient orientation into 9 bins (0-180)

• The vote is the gradient magnitude
Histogram of Oriented Gradients (HOG)

• Step 4: Concatenate the four histograms of each block.
Histogram of Oriented Gradients (HOG)

Let vector $v$ be concatenation of the four histograms from step 4.

- **Step 5: Normalize $v$.**

Here we have three options for how to do it:

- **Option 1:** Divide $v$ by its Euclidean norm.

- **Option 2:** Divide $v$ by its L1 norm (the L1 norm is the sum of all absolute values of $v$).

- **Option 3:**
  - Divide $v$ by its Euclidean norm.
  - In the resulting vector, clip any value over 0.2
  - Then, renormalize the resulting vector by dividing again by its Euclidean norm.
Summary of HOG computation

• Step 1: Extract a square window (called “block”) of some size around the pixel location of interest.
• Step 2: Divide block into a square grid of sub-blocks (called “cells”) (2x2 grid in our example, resulting in four cells).
• Step 3: Compute orientation histogram of each cell.
• Step 4: Concatenate the four histograms.
• Step 5: normalize \( v \) using one of the three options described previously.
Histogram of Oriented Gradients (HOG)

• Parameters and design options:
  • Angles range from 0 to 180 or from 0 to 360 degrees?
    • In the Dalal & Triggs paper, a range of 0 to 180 degrees is used, and
    • HOGs are used for detection of pedestrians.

• Number of orientation bins.
  • Usually 9 bins, each bin covering 20 degrees.

• Cell size.
  • Cells of size 8x8 pixels are often used.

• Block size.
  • Blocks of size 2x2 cells (16x16 pixels) are often used.

• Usually a HOG feature has 36 dimensions.
  • 4 cells * 9 orientation bins.
Histogram of Oriented Gradients (HOG)

Input image

Histogram of Oriented Gradients
Scale Invariant Feature Transform (SIFT)

A feature detector and a feature descriptor
Scale Invariant Feature Transform (SIFT)

- Lowe, D. 2004, IJCV
Scale Invariant Feature Transform (SIFT)

• Image content is transformed into local feature coordinates

• Invariant to
  • translation
  • rotation
  • scale, and
  • other imaging parameters
Scale Invariant Feature Transform (SIFT)

- Image content is transformed into local feature coordinates
Scale Invariant Feature Transform (SIFT)

• Procedure at High Level

- **Scale-Space Extrema Detection**: Search over multiple scales and image locations.
- **Keypoint Localization**: Fit a model to determine location and scale. Select keypoints based on a measure of stability.
- **Orientation Assignment**: Compute best orientation(s) for each keypoint region.
- **Keypoint Description**: Use local image gradients at selected scale and rotation to describe each keypoint region.
SIFT. Automatic scale selection

\[ f(I_{i_1...i_m}(x,\sigma)) = f(I_{i_1...i_m}(x',\sigma')) \]

How to find patch sizes at which \( f \) response is equal?

What is a good \( f \)?
SIFT. Automatic scale selection

Function responses for increasing scale (scale signature)
SIFT. Automatic scale selection

Function responses for increasing scale (scale signature)
SIFT. Automatic scale selection

Function responses for increasing scale (scale signature)
SIFT. Automatic scale selection

Function responses for increasing scale (scale signature)
SIFT. Automatic scale selection

Function responses for increasing scale (scale signature)
SIFT. Automatic scale selection

Function responses for increasing scale (scale signature)
What is a useful signature function $f$?
Blob detection
Formally…

Highest response when the signal has the same characteristic scale as the filter
What is a useful signature function $f$?

“Blob” detector is common for corners
- Laplacian ($2^{nd}$ derivative) of Gaussian (LoG)

![Graph showing scale space, function response, and image blob size.]
Find local maxima in position-scale space
What happens if you apply different Laplacian filters?
What happened when you applied different Laplacian filters?
optimal scale

2.1  4.2  6.0  9.8  15.5  17.0

Full size image

2.1  4.2  6.0  9.8  15.5  17.0

3/4 size image
optimal scale

Full size image

3/4 size image
Scale Invariant Detection

- Functions for determining scale \( f = \text{Kernel} \ast \text{Image} \)

Kernels:

\[
\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}
\]

(Laplacian)

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

(Difference of Gaussians)

where Gaussian

\[
G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2 + y^2}{2\sigma^2}}
\]

Note: The LoG and DoG operators are both rotation equivariant
Alternative to compute Laplacian of Gaussian

• Approximate LoG with Difference-of-Gaussian (DoG).

1. Blur image with $\sigma$ Gaussian kernel
2. Blur image with $k\sigma$ Gaussian kernel
3. Subtract 2. from 1.
Scale-Space
Find local maxima in position-scale space of DoG

Find maxima/minima

⇒ List of (x, y, s)
Results: Difference of Gaussians

- Larger circles = larger scale
- Descriptors with maximal scale response
SIFT Orientation estimation

• Compute gradient orientation histogram
• Select dominant orientation $\Theta$
SIFT Orientation Normalization

• Compute gradient orientation histogram
• Select dominant orientation $\Theta$
• Normalize: rotate to fixed orientation
SIFT Detector

• In addition to position x, y of the feature,
  • Scale $\sigma$ (determined by smoothing value)
  • Orientation of dominant gradient $\theta$
SIFT descriptor

• Compute on local 16 x 16 window around detection.
• Rotate and scale window according to discovered orientation $\Theta$ and scale $\sigma$ (gain invariance).
• Compute gradients weighted by a Gaussian of variance half the window (for smooth falloff).
SIFT descriptor

• 4x4 array of gradient orientation histograms weighted by gradient magnitude.
• Bin into 8 orientations x 4x4 array = 128 dimensions.
SIFT Descriptor Extraction

Gradient magnitude and orientation

128 dimensional vector

8 bin ‘histogram’ - add magnitude amounts!

Key Point

Utkarsh Sinha
Reduce effect of illumination

- 128-dim vector normalized to 1
- Threshold gradient magnitudes to avoid excessive influence of high gradients
  - After normalization, clamp gradients > 0.2
  - Renormalize
Review: Local Descriptors

• Most features can be thought of as
  • templates,
  • histograms (counts),
  • or combinations

• The ideal descriptor should be
  — Robust and Distinctive
  — Compact and Efficient

• Most available descriptors focus on edge/gradient information
  — Capture texture information
  — Color rarely used
References

Basic reading:
• Szeliski textbook, Sections 4.1.
Questions?