Detecting corners
Course announcements

• Homework 1 posted on course website.
  - Due on February 5\textsuperscript{th} at 23:59.
  - This homework is in Matlab.
  - How many of you have looked at/started/finished homework 1?

• First theory quiz on course website and due on February 3\textsuperscript{rd}, at 23:59.

• From here on, all office hours will be at Smith Hall 200.
  - Conference room next to the second floor restrooms.
Overview of today’s lecture

Leftover from Lecture 4:
- More on Hough lines.
- Hough circles.

New in lecture 5:
- Why detect corners?
- Visualizing quadratics.
- Harris corner detector.
- Multi-scale detection.
- Multi-scale blob detection.
Slide credits

Most of these slides were adapted from:

• Kris Kitani (15-463, Fall 2016).

Some slides were inspired or taken from:

• Fredo Durand (MIT).
• James Hays (Georgia Tech).
Why detect corners?
Why detect corners?

- Image alignment (homography, fundamental matrix)
- 3D reconstruction
- Motion tracking
- Object recognition
- Indexing and database retrieval
- Robot navigation
Planar object instance recognition

Database of planar objects

Instance recognition
3D object recognition

Database of 3D objects

3D objects recognition
Location Recognition
Robot Localization
Image matching
Where are the corresponding points?
Pick a point in the image.
Find it again in the next image.

*What type of feature would you select?*
Pick a point in the image. Find it again in the next image.

What type of feature would you select?
Pick a point in the image. Find it again in the next image.

What type of feature would you select?
a corner
Visualizing quadratics
Equation of a circle

\[ 1 = x^2 + y^2 \]

Equation of a ‘bowl’ (paraboloid)

\[ f(x, y) = x^2 + y^2 \]

*If you slice the bowl at*

\[ f(x, y) = 1 \]

*what do you get?*
Equation of a circle

\[ 1 = x^2 + y^2 \]

Equation of a ‘bowl’ (paraboloid)

\[ f(x, y) = x^2 + y^2 \]

If you slice the bowl at

\[ f(x, y) = 1 \]

what do you get?
\[ f(x, y) = x^2 + y^2 \]

can be written in matrix form like this...

\[
\begin{bmatrix}
  x \\
  y
\end{bmatrix}
= 
\begin{bmatrix}
    1 & 0 \\
    0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y
\end{bmatrix}
\]
\[ f(x, y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \]

‘sliced at 1’
What happens if you *increase* coefficient on $x$?

\[
f(x, y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]

and slice at 1
What happens if you **increase** coefficient on \( x \)?

\[
f(x, y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]

and slice at 1

decrease width in \( x \)!
What happens if you *increase* coefficient on \( y \)?

\[
    f(x, y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]

and slice at 1
What happens if you _increase_ coefficient on \( y \) and slice at 1

\[
f(x, y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]

and slice at 1
can be written in matrix form like this…

\[ f(x, y) = x^2 + y^2 \]

\[ f(x, y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \]

**What’s the shape?**

**What are the eigenvectors?**

**What are the eigenvalues?**
\[ f(x, y) = x^2 + y^2 \]

can be written in matrix form like this...

\[
f(x, y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]

Result of Singular Value Decomposition (SVD)

\[
\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}^T
\]

- eigenvectors
- eigenvalues along diagonal
- axis of the 'ellipse slice'
- Inverse sqsr of length of the quadratic along the axis
$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}^T$$
Recall:

\[
f(x, y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]

you can smash this bowl in the \textbf{y} direction

\[
f(x, y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]

you can smash this bowl in the \textbf{x} direction

\[
f(x, y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]
\[ A = \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}^T \]
\[ A = \begin{bmatrix} 3.25 & 1.30 \\ 1.30 & 1.75 \end{bmatrix} = \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix} \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix}^T \]
\[ A = \begin{bmatrix} 7.75 & 3.90 \\ 3.90 & 3.25 \end{bmatrix} = \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 10 \end{bmatrix} \begin{bmatrix} 0.50 & -0.87 \\ -0.87 & -0.50 \end{bmatrix}^T \]
We will need this to understand the...

**Error function for Harris Corners**

The surface $E(u,v)$ is locally approximated by a quadratic form

\[ E(u,v) \approx [u \ v] \ M \begin{bmatrix} u \\ v \end{bmatrix} \]

\[ M = \sum \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \]
Harris corner detector
How do you find a corner?
How do you find a corner?

Easily recognized by looking through a small window

Shifting the window should give large change in intensity
Easily recognized by looking through a small window
Shifting the window should give large change in intensity

“flat” region: no change in all directions
“edge”: no change along the edge direction
“corner”: significant change in all directions

[Moravec 1980]
Design a program to detect corners
(hint: use image gradients)
Finding corners
(a.k.a. PCA)

1. Compute image gradients over small region

2. Subtract mean from each image gradient

3. Compute the covariance matrix

4. Compute eigenvectors and eigenvalues

\[
\begin{bmatrix}
\sum_{p \in P} I_x I_x & \sum_{p \in P} I_x I_y \\
\sum_{p \in P} I_x I_y & \sum_{p \in P} I_y I_y
\end{bmatrix}
\]

5. Use threshold on eigenvalues to detect corners
1. Compute image gradients over a small region
(not just a single pixel)
1. Compute image gradients over a small region
   (not just a single pixel)

\[ I_x = \frac{\partial I}{\partial x} \]

\[ I_y = \frac{\partial I}{\partial y} \]
visualization of gradients

image

X derivative

Y derivative
What does the distribution tell you about the region?
distribution reveals edge orientation and magnitude
How do you quantify orientation and magnitude?
2. Subtract the mean from each image gradient
2. Subtract the mean from each image gradient

constant intensity gradient

plot intensities

intensities along the line
2. Subtract the mean from each image gradient

plot of image gradients

constant intensity gradient

plot intensities

intensities along the line

\[ I_y = \frac{\partial I}{\partial y} \]

\[ I_x = \frac{\partial I}{\partial x} \]

subtract mean
2. Subtract the mean from each image gradient

\[ I_y = \frac{\partial I}{\partial y} \]
\[ I_x = \frac{\partial I}{\partial x} \]

plot of image gradients

subtract mean

plot intensities

intensities along the line

constant intensity gradient

data is centered (‘DC’ offset is removed)
3. Compute the covariance matrix
3. Compute the covariance matrix

$$\begin{bmatrix}
\sum_{p \in P} I_x I_x & \sum_{p \in P} I_x I_y \\
\sum_{p \in P} I_y I_x & \sum_{p \in P} I_y I_y
\end{bmatrix}$$

\[
\sum_{p \in P} I_x I_y = \text{sum} (\text{array of } x \text{ gradients} \cdot \text{array of } y \text{ gradients})
\]

**Where does this covariance matrix come from?**
Easily recognized by looking through a small window

Shifting the window should give large change in intensity

“flat” region: no change in all directions

“edge”: no change along the edge direction

“corner”: significant change in all directions

[Moravec 1980]
Error function

Change of intensity for the shift \([u,v]\):

\[
E(u,v) = \sum_{x,y} w(x,y) \left[ I(x+u, y+v) - I(x, y) \right]^2
\]

Window function \(w(x,y)\):
- 1 in window, 0 outside
- Gaussian

Some mathematical background…
Error function approximation

Change of intensity for the shift \([u, v]\):

\[
E(u, v) = \sum_{x, y} w(x, y)[I(x + u, y + v) - I(x, y)]^2
\]

First-order Taylor expansion of \(I(x, y)\) about \((0,0)\)
(bilinear approximation for small shifts)
Bilinear approximation

For small shifts \([u, v]\) we have a ‘bilinear approximation’:

\[
E(u, v) \approx \begin{bmatrix} u \\ v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}
\]

where M is a 2×2 matrix computed from image derivatives:

\[
M = \sum_{x,y} w(x, y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}
\]

Change in appearance for a shift \([u,v]\)

‘second moment’ matrix ‘structure tensor’
By computing the gradient covariance matrix...

\[
\begin{bmatrix}
\sum_{p \in P} I_x I_x & \sum_{p \in P} I_x I_y \\
\sum_{p \in P} I_y I_x & \sum_{p \in P} I_y I_y
\end{bmatrix}
\]

we are fitting a quadratic to the gradients over a small image region.
Visualization of a quadratic

The surface $E(u,v)$ is locally approximated by a quadratic form

\[ E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} \]

\[ M = \sum \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \]
Which error surface indicates a good image feature?

What kind of image patch do these surfaces represent?
Which error surface indicates a good image feature?

flat
Which error surface indicates a good image feature?

- flat
- edge
- ‘line’
Which error surface indicates a good image feature?

flat

edge
‘line’

corner
‘dot’
4. Compute eigenvalues and eigenvectors
4. Compute eigenvalues and eigenvectors

\[ Me = \lambda e \]

\[(M - \lambda I)e = 0\]
1. Compute the determinant of $M - \lambda I$ (returns a polynomial)

4. Compute eigenvalues and eigenvectors

$$Me = \lambda e$$

$$(M - \lambda I)e = 0$$
1. Compute the determinant of \( M - \lambda I \) (returns a polynomial)

2. Find the roots of polynomial \( \det(M - \lambda I) = 0 \) (returns eigenvalues)

4. Compute eigenvalues and eigenvectors

\[ Me = \lambda e \]

\[ (M - \lambda I)e = 0 \]
4. Compute eigenvalues and eigenvectors

\[ Me = \lambda e \]

\[(M - \lambda I)e = 0\]

1. Compute the determinant of \(M - \lambda I\)
   (returns a polynomial)

2. Find the roots of polynomial \(\det(M - \lambda I) = 0\)
   (returns eigenvalues)

3. For each eigenvalue, solve \((M - \lambda I)e = 0\)
   (returns eigenvectors)
eig(M)
Visualization as an ellipse

Since $M$ is symmetric, we have

$$M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

We can visualize $M$ as an ellipse with axis lengths determined by the eigenvalues and orientation determined by $R$.

Ellipse equation:

$$\begin{bmatrix} u \\ v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$$

$\lambda_{\text{max}}^{-1/2}$

$\lambda_{\text{min}}^{-1/2}$

direction of the fastest change

direction of the slowest change
interpreting eigenvalues

What kind of image patch does each region represent?

\( \lambda_2 \gg \lambda_1 \)

\( \lambda_1 \approx 0 \)
\( \lambda_2 \approx 0 \)

\( \lambda_1 \gg \lambda_2 \)
interpreting eigenvalues

\[ \lambda_2 \gg \lambda_1 \]

- 'horizontal' edge
- 'vertical' edge
- flat
- corner

\[ \lambda_1 \sim \lambda_2 \]

\[ \lambda_1 \gg \lambda_2 \]
interpreting eigenvalues

\[ \lambda_2 \gg \lambda_1 \]

- 'horizontal' edge
- 'vertical' edge
- corner
- flat

\[ \lambda_1 \sim \lambda_2 \]

\[ \lambda_1 \gg \lambda_2 \]
interpreting eigenvalues

- \( \lambda_2 \gg \lambda_1 \) (horizontal edge)
- \( \lambda_2 \sim \lambda_1 \) (flat)
- \( \lambda_1 \gg \lambda_2 \) (vertical edge)
- Corner
5. Use threshold on eigenvalues to detect corners
5. Use threshold on eigenvalues to detect corners

Think of a function to score ‘cornerness’
5. Use threshold on eigenvalues to detect corners

Think of a function to score ‘cornerness’
5. Use threshold on eigenvalues to detect corners

\[ R = \min(\lambda_1, \lambda_2) \]
5. Use threshold on eigenvalues to detect corners
   (a function of $\lambda$)

Eigenvalues need to be bigger than one.

$$R = \lambda_1 \lambda_2 - \kappa (\lambda_1 + \lambda_2)^2$$

Can compute this more efficiently…
5. Use threshold on eigenvalues to detect corners

(a function of )

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

\[ \begin{align*}
R &< 0 & R &> 0 \\
R &\ll 0 & R &< 0 \\
\end{align*} \]

\[
\det M = \lambda_1 \lambda_2
\]

\[
\text{trace } M = \lambda_1 + \lambda_2
\]

\[
\det \left( \begin{bmatrix} a & b \\ c & d \end{bmatrix} \right) = ad - bc
\]

\[
\text{trace } \left( \begin{bmatrix} a & b \\ c & d \end{bmatrix} \right) = a + d
\]
Harris & Stephens (1988)

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

Kanade & Tomasi (1994)

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

Nobel (1998)

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

1. Compute x and y derivatives of image

\[ I_x = G^x_\sigma \ast I \quad I_y = G^y_\sigma \ast I \]

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

\[ S_{x^2} = G^{\sigma'} \ast I_{x^2} \quad S_{y^2} = G^{\sigma'} \ast I_{y^2} \quad S_{xy} = G^{\sigma'} \ast I_{xy} \]
Harris Detector


4. Define the matrix at each pixel

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

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.
Yet another option: \[ f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2} \]

How do you write this equivalently using determinant and trace?
Yet another option:

\[ f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2} = \frac{\text{determinant}(H)}{\text{trace}(H)} \]
Different criteria
Corner response
Thresholded corner response
Non-maximal suppression
Harris corner response is invariant to rotation

Ellipse rotates but its shape (eigenvalues) remains the same

Corner response $R$ is invariant to image rotation
Harris corner response is invariant to intensity changes

Partial invariance to **affine intensity** change

- Only derivatives are used $\Rightarrow$ invariance to intensity shift $I \rightarrow I + b$

- Intensity scale: $I \rightarrow aI$
The Harris detector is not invariant to changes in ...
The Harris corner detector is not invariant to scale.
Multi-scale detection
How can we make a feature detector scale-invariant?
How can we automatically select the scale?
Multi-scale blob detection
Intuitively...

Find local maxima in both **position** and **scale**
Formally…

Highest response when the signal has the same **characteristic scale** as the filter.
characteristic scale - the scale that produces peak filter response

we need to search over characteristic scales
What happens if you apply different Laplacian filters?
jet color scale
blue: low, red: high
What happened when you applied different Laplacian filters?
peak!

sigma=9.8
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

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
How would you implement scale selection?
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 \((x, y, s)\)
References

Basic reading:
• Szeliski textbook, Sections 4.1.