Feature extraction: Corners and blobs
Review: Linear filtering and edge detection

- Name two different kinds of image noise
- Name a non-linear smoothing filter
- What advantages does median filtering have over Gaussian smoothing?
- What is aliasing?
- How do we find edges?
- Why do we need to smooth before computing image derivatives?
- What are some characteristics of an “optimal” edge detector?
- What is nonmaximum suppression?
- What is hysteresis thresholding?
Why extract features?

• Motivation: panorama stitching
  • We have two images – how do we combine them?
Why extract features?

- **Motivation: panorama stitching**
  - We have two images – how do we combine them?

Step 1: extract features
Step 2: match features
Why extract features?

• Motivation: panorama stitching
  • We have two images – how do we combine them?

Step 1: extract features
Step 2: match features
Step 3: align images
Characteristics of good features

• Repeatability
  • The same feature can be found in several images despite geometric and photometric transformations

• Saliency
  • Each feature has a distinctive description

• Compactness and efficiency
  • Many fewer features than image pixels

• Locality
  • A feature occupies a relatively small area of the image; robust to clutter and occlusion
Applications

Feature points are used for:

- Motion tracking
- Image alignment
- 3D reconstruction
- Object recognition
- Indexing and database retrieval
- Robot navigation
Finding Corners

• Key property: in the region around a corner, image gradient has two or more dominant directions

• Corners are repeatable and distinctive

The 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: Mathematics

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

Shifted intensity

Intensity

Window function \(w(x, y) =

1 \text{ in window, } 0 \text{ outside}

or

Gaussian

Source: R. Szeliski
Harris Detector: Mathematics

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

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

Second-order Taylor expansion of \(E(u,v)\) about \((0,0)\) (bilinear approximation for small shifts):

\[
E(u,v) \approx E(0,0) + [u \ v] \begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2} [u \ v] \begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}
\]
Harris Detector: Mathematics

The bilinear approximation simplifies to

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

where \( M \) is a \( 2 \times 2 \) matrix computed from image derivatives:

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

\[ M = \begin{bmatrix} \frac{1}{\sum I_x I_x} & \frac{1}{\sum I_x I_y} \\ \frac{1}{\sum I_x I_y} & \frac{1}{\sum I_y I_y} \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \ I_y] = \sum \nabla I (\nabla I)^T \]
Interpreting the second moment matrix

First, consider an axis-aligned corner:
First, consider an axis-aligned corner:

\[ M = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \]

This means dominant gradient directions align with x or y axis

If either \( \lambda \) is close to 0, then this is not a corner, so look for locations where both are large.
General Case

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:

$$\begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$$
Visualization of second moment matrices
Visualization of second moment matrices
Interpreting the eigenvalues

Classification of image points using eigenvalues of $M$:

- **Corner**
  - $\lambda_1$ and $\lambda_2$ are large,
  - $\lambda_1 \sim \lambda_2$;
  - $E$ increases in all directions

- **Edge**
  - $\lambda_2 \gg \lambda_1$

- **Flat** region
  - $\lambda_1$ and $\lambda_2$ are small;
  - $E$ is almost constant in all directions
Corner response function

\[ R = \det(M) - \alpha \text{trace}(M)^2 = \lambda_1 \lambda_2 - \alpha (\lambda_1 + \lambda_2)^2 \]

\( \alpha: \text{constant (0.04 to 0.06)} \)
Harris Detector: Steps
Harris Detector: Steps

Compute corner response $R$
Harris Detector: Steps

Find points with large corner response: $R > \text{threshold}$
Harris Detector: Steps

Take only the points of local maxima of $R$
Harris Detector: Steps
Harris detector: Summary of steps

1. Compute Gaussian derivatives at each pixel
2. Compute second moment matrix $M$ in a Gaussian window around each pixel
3. Compute corner response function $R$
4. Threshold $R$
5. Find local maxima of response function (nonmaximum suppression)
Invariance

• We want features to be detected despite geometric or photometric changes in the image: if we have two transformed versions of the same image, features should be detected in corresponding locations.
Models of Image Change

Geometric
- Rotation
- Scale
- Affine
  valid for: orthographic camera, locally planar object

Photometric
- Affine intensity change \((l \rightarrow a l + b)\)
Harris Detector: Invariance Properties

Rotation

Ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner response $R$ is invariant to image rotation
Harris Detector: Invariance Properties

Affine intensity change

✓ Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$
✓ Intensity scale: $I \rightarrow a I$

Partially invariant to affine intensity change
Harris Detector: Invariance Properties

Scaling

Corner

All points will be classified as edges

Not invariant to scaling
Scale-invariant feature detection

- Goal: independently detect corresponding regions in scaled versions of the same image
- Need *scale selection* mechanism for finding characteristic region size that is *covariant* with the image transformation
Scale-invariant features: Blobs
Recall: Edge detection

Edge = maximum of derivative

Source: S. Seitz
Edge detection, Take 2

Edge = zero crossing of second derivative of Gaussian (Laplacian)

Source: S. Seitz
From edges to blobs

- Edge = ripple
- Blob = superposition of two ripples

**Spatial selection**: the magnitude of the Laplacian response will achieve a maximum at the center of the blob, provided the scale of the Laplacian is “matched” to the scale of the blob.
Scale selection

• We want to find the characteristic scale of the blob by convolving it with Laplacians at several scales and looking for the maximum response.

• However, Laplacian response decays as scale increases:

Why does this happen?
Scale normalization

- The response of a derivative of Gaussian filter to a perfect step edge decreases as $\sigma$ increases.
Scale normalization

- The response of a derivative of Gaussian filter to a perfect step edge decreases as $\sigma$ increases.
- To keep response the same (scale-invariant), must multiply Gaussian derivative by $\sigma$.
- Laplacian is the second Gaussian derivative, so it must be multiplied by $\sigma^2$. 
Effect of scale normalization

Original signal

Unnormalized Laplacian response

Scale-normalized Laplacian response

maximum
Blob detection in 2D

Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D

\[ \nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \]
Blob detection in 2D

Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D

Scale-normalized:

\[ \nabla^2_{\text{norm}} g = \sigma^2 \left( \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \right) \]
Scale selection

• The 2D Laplacian is given by

\[(x^2 + y^2 - 2\sigma^2) e^{-(x^2+y^2)/2\sigma^2} \text{ (up to scale)}\]

• Therefore, for a binary circle of radius r, the Laplacian achieves a maximum at \(\sigma = r / \sqrt{2}\)
Characteristic scale

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

Scale-space blob detector

1. Convolve image with scale-normalized Laplacian at several scales
2. Find maxima of squared Laplacian response in scale-space
Scale-space blob detector: Example
Scale-space blob detector: Example

sigma = 11.9912
Scale-space blob detector: Example
Efficient implementation

Approximating 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)
From scale invariance to affine invariance
Affine adaptation

Recall: \( M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R \)

We can visualize \( M \) as an ellipse with axis lengths determined by the eigenvalues and orientation determined by \( R \)

Ellipse equation:

\[
\begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}
\]
Affine adaptation

- The second moment ellipse can be viewed as the “characteristic shape” of a region
- We can normalize the region by transforming the ellipse into a unit circle
Orientation ambiguity

- There is no unique transformation from an ellipse to a unit circle
  - We can rotate or flip a unit circle, and it still stays a unit circle
Orientation ambiguity

• There is no unique transformation from an ellipse to a unit circle
  • We can rotate or flip a unit circle, and it still stays a unit circle

• So, to assign a unique orientation to keypoints:
  • Create histogram of local gradient directions in the patch
  • Assign canonical orientation at peak of smoothed histogram
Affine adaptation

- Problem: the second moment “window” determined by weights $w(x,y)$ must match the characteristic shape of the region

- Solution: iterative approach
  - Use a circular window to compute second moment matrix
  - Perform affine adaptation to find an ellipse-shaped window
  - Recompute second moment matrix using new window and iterate
Iterative affine adaptation


http://www.robots.ox.ac.uk/~vgg/research/affine/
Affine adaptation example

Scale-invariant regions (blobs)
Affine adaptation example

Affine-adapted blobs
Summary: Feature extraction

Extract affine regions → Normalize regions → Eliminate rotational ambiguity → Compute appearance descriptors

SIFT (Lowe '04)
Invariance vs. covariance

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

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

Covariant detection => invariant description
Assignment 1 due February 14

Implement the Laplacian blob detector:

http://www.cs.unc.edu/~lazebnik/spring08/assignment1.html
Next time: Fitting