Recap: edge detection
Canny edge detector

1. Filter image with $x$, $y$ derivatives of Gaussian
2. Find magnitude and orientation of gradient
3. Non-maximum suppression:
   - Thin multi-pixel wide “ridges” down to single pixel width
4. Thresholding and linking (hysteresis):
   - Define two thresholds: low and high
   - Use the high threshold to start edge curves and the low threshold to continue them

• MATLAB: edge(image, ‘canny’)
Hysteresis thresholding

• Check that maximum value of gradient value is sufficiently large
  – drop-outs? use **hysteresis**
    • use a high threshold to start edge curves and a low threshold to continue them.
To be continued

• How do we know that one edge detector is better than another?
Interest Points and Corners

Read Szeliski 4.1

Computer Vision

James Hays
Correspondence across views

• Correspondence: matching points, patches, edges, or regions across images
Example: estimating “fundamental matrix” that corresponds two views
Example: structure from motion
Applications

- Feature points are used for:
  - Image alignment
  - 3D reconstruction
  - Motion tracking
  - Robot navigation
  - Indexing and database retrieval
  - Object recognition
This class: interest points and local features

• Note: “interest points” = “keypoints”, also sometimes called “features”
This class: interest points

• Suppose you have to click on some point, go away and come back after I deform the image, and click on the same points again.
  – Which points would you choose?
Overview of Keypoint Matching

1. Find a set of distinctive keypoints
2. Define a region around each keypoint
3. Compute a local descriptor from the normalized region
4. Match local descriptors

\[ d(f_A, f_B) < T \]
Goals for Keypoints

Detect points that are *repeatable* and *distinctive*
Invariant Local Features

Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters.
Why extract features?

- Motivation: panorama stitching
  - We have two images – how do we combine them?
Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.

\[ \mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}] \]

\[ \mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}] \]

3) Matching: Determine correspondence between descriptors in two views

Kristen Grauman
Characteristics of good features

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

• **Saliency**
  • Each feature is distinctive

• **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
Goal: interest operator repeatability

• We want to detect (at least some of) the same points in both images.

No chance to find true matches!

• Yet we have to be able to run the detection procedure *independently* per image.
Goal: descriptor distinctiveness

- We want to be able to reliably determine which point goes with which.

- Must provide some invariance to geometric and photometric differences between the two views.

Kristen Grauman
Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.

3) Matching: Determine correspondence between descriptors in two views
Many Existing Detectors Available

**Hessian & Harris**
[Harris ‘88], [Beaudet ‘78]

**Laplacian, DoG**
[Lindeberg ‘98], [Lowe 1999]

**Harris-/Hessian-Laplace**
[Mikolajczyk & Schmid ‘04]

**Harris-/Hessian-Affine**
[Mikolajczyk & Schmid ‘01]

**EBR and IBR**
[Tuytelaars & Van Gool ‘04]

**MSER**
[Matas ‘02]

**Salient Regions**
[Kadir & Brady ‘01]

**Others...**
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
Corner Detection: Mathematics

Change in appearance of window \( w(x,y) \) for the shift \([u,v]\):

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

Change in appearance of window \( w(x,y) \) for the shift \([u,v]\):

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

Change in appearance of window \( w(x,y) \) 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 in window, 0 outside
- or
- Gaussian

Source: R. Szeliski
Corner Detection: Mathematics

Change in appearance of window $w(x, y)$ for the shift $[u, v]$: 

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

We want to find out how this function behaves for small shifts

$E(u, v)$
Corner Detection: Mathematics

Change in appearance of window $w(x,y)$ for the shift $[u,v]$:

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

We want to find out how this function behaves for small shifts.

But this is very slow to compute naively. $O(\text{window}_\text{width}^2 \times \text{shift}_\text{range}^2 \times \text{image}_\text{width}^2)$

$O(\ 11^2 \times 11^2 \times 600^2\ ) = 5.2$ billion of these

14.6 thousand per pixel in your image
Corner Detection: Mathematics

Change in appearance of window \( w(x,y) \) for the shift \([u,v]\):

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

We want to find out how this function behaves for small shifts.

Recall Taylor series expansion. A function \( f \) can be approximated at point \( a \) as

\[
f(a) + \frac{f'(a)}{1!}(x - a) + \frac{f''(a)}{2!}(x - a)^2 + \frac{f'''(a)}{3!}(x - a)^3 + \cdots.
\]
Recall: Taylor series expansion

A function $f$ can be approximated as

$$f(a) + \frac{f'(a)}{1!}(x - a) + \frac{f''(a)}{2!}(x - a)^2 + \frac{f'''(a)}{3!}(x - a)^3 + \cdots.$$ 

Approximation of $f(x) = e^x$ centered at $f(0)$
Corner Detection: Mathematics

Change in appearance of window \( w(x,y) \) for the shift \([u,v]\):

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

We want to find out how this function behaves for small shifts.

Local quadratic approximation of \( E(u,v) \) in the neighborhood of \((0,0)\) is given by the second-order Taylor expansion:

\[
E(u, v) \approx E(0,0) + [u \ v] \begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2} [u \ v] \begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix} [u \ v]
\]
Local quadratic approximation of $E(u, v)$ in the neighborhood of $(0,0)$ is given by the second-order Taylor expansion:

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

Always 0

First derivative is 0

$E(u, v)$
Corner Detection: Mathematics

The quadratic approximation simplifies to

\[ E(u, v) \approx [u \ v] M [u \v] \]

where \( M \) is a *second moment matrix* computed from image derivatives:

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

\[
M = \begin{bmatrix}
\sum I_xI_x & \sum I_xI_y \\
\sum I_xI_y & \sum I_yI_y
\end{bmatrix} = \sum \begin{bmatrix}
I_x \\
I_y
\end{bmatrix} [I_x \ I_y] = \sum \nabla I(\nabla I)^T
\]
Corners as distinctive interest points

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

2 x 2 matrix of image derivatives (averaged in neighborhood of a point).

Notation:

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

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

\[ I_x I_y \leftrightarrow \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \]
Interpreting the second moment matrix

The surface $E(u, v)$ is locally approximated by a quadratic form. Let’s try to understand its shape.

$$E(u, v) \approx [u \ v] \ M \ [u \ v]$$

$$M = \sum_{x, y} w(x, y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$
Consider a horizontal “slice” of $E(u, v)$: $\begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$

This is the equation of an ellipse.
Consider a horizontal “slice” of $E(u, v)$: $\begin{bmatrix} u \\ v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$

This is the equation of an ellipse.

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

The axis lengths of the ellipse are determined by the eigenvalues and the orientation is determined by $R$. 

$\lambda_{\text{max}}^{-1/2}$ and $\lambda_{\text{min}}^{-1/2}$ represent the direction of the fastest and slowest change, respectively.
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

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

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

\( \alpha: \) constant (0.04 to 0.06)
1) Compute $M$ matrix for each image window to get their *cornerness* scores.

2) Find points whose surrounding window gave large corner response ($f >$ threshold)

3) Take the points of local maxima, i.e., perform non-maximum suppression

Harris Detector [Harris88]

- Second moment matrix

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

1. Image derivatives (optionally, blur first)

\[
det M = \lambda_1 \lambda_2
\]

\[
trace M = \lambda_1 + \lambda_2
\]

2. Square of derivatives

3. Gaussian filter \(g(\sigma_I)\)

4. Cornerness function – both eigenvalues are strong

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

5. Non-maxima suppression
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
Invariance and covariance

• We want corner locations to be *invariant* to photometric transformations and *covariant* to geometric transformations
  • **Invariance**: image is transformed and corner locations do not change
  • **Covariance**: if we have two transformed versions of the same image, features should be detected in corresponding locations
Affine intensity change

- Only derivatives are used => invariance to intensity shift \( I \rightarrow I + b \)
- Intensity scaling: \( I \rightarrow a \, I \)

Partially invariant to affine intensity change
Image translation

- Derivatives and window function are shift-invariant

Corner location is covariant w.r.t. translation
Second moment ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner location is covariant w.r.t. rotation
Scaling

Corner location is not covariant to scaling!

All points will be classified as edges