Feature Detectors and Descriptors: Corners, Blobs, and SIFT
Why Extract Keypoints?

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

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

**Step 1:** extract keypoints

**Step 2:** match keypoint features
Why Extract Keypoints?

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

**Step 1:** extract keypoints
**Step 2:** match keypoint features
**Step 3:** align images
Characteristics of Good Keypoints

• **Repeatability**
  – Can be found despite geometric and photometric transformations

• **Salience**
  – Each keypoint is distinctive

• **Compactness and efficiency**
  – Many fewer keypoints than image pixels

• **Locality**
  – Occupies small area of the image; robust to clutter and occlusion
Applications

• Keypoints are used for:
  – Image alignment
  – 3D reconstruction
  – Motion tracking
  – Robot navigation
  – Indexing and database retrieval
  – Object recognition
Kinds of Keypoints

• Corners

• Blobs
Edges vs. Corners

- Edges = maxima in intensity gradient
Corners = lots of variation in direction of gradient in a small neighborhood
Detecting Corners

• How to detect this variation?
• Not enough to check average $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$
Detecting Corners

• **Claim**: the following “structure” matrix summarizes the second-order statistics of the gradient

\[
C = \left[ \begin{array}{cc}
\sum f_x^2 & \sum f_x f_y \\
\sum f_x f_y & \sum f_y^2 \\
\end{array} \right]
\]

\[f_x = \frac{\partial f}{\partial x}, f_y = \frac{\partial f}{\partial y}\]

• Summations over local neighborhoods
  – Can have spatially-varying weights (Gaussian, etc.)
Detecting Corners

- Examine behavior of $C$ by testing its effect in simple cases
- Case #1: Single edge in local neighborhood
Case#1: Single Edge

- Let \((a,b)\) be gradient along edge
- Compute \(C \cdot (a,b)\):

\[
C \cdot \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum f_x^2 & \sum f_x f_y \\ \sum f_x f_y & \sum f_y^2 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \\
= \sum (\nabla f)(\nabla f)^T \begin{bmatrix} a \\ b \end{bmatrix} \\
= \sum (\nabla f) (\nabla f \cdot \begin{bmatrix} a \\ b \end{bmatrix})
\]
Case #1: Single Edge

- However, in this simple case, the only nonzero terms are those where $\nabla f = (a,b)$
- So, $C \cdot (a,b)$ is just some multiple of $(a,b)$
Case #2: Corner

- Assume there is a corner, with perpendicular gradients \((a, b)\) and \((c, d)\)
Case #2: Corner

- What is $C \cdot (a,b)$?
  - Since $(a,b) \cdot (c,d) = 0$, the only nonzero terms are those where $\nabla f = (a,b)$
  - So, $C \cdot (a,b)$ is again just a multiple of $(a,b)$

- What is $C \cdot (c,d)$?
  - Since $(a,b) \cdot (c,d) = 0$, the only nonzero terms are those where $\nabla f = (c,d)$
  - So, $C \cdot (c,d)$ is a multiple of $(c,d)$
Corner Detection

• Matrix times vector = multiple of vector
• Eigenvectors and eigenvalues!
• In particular, if C has one large eigenvalue, there’s an edge
• If C has two large eigenvalues, have corner
• “Harris” corner detector
  – Harris & Stephens 1988 look at trace and determinant of C; Shi & Tomasi 1994 directly look at minimum eigenvalue
Visualization of Structure Matrix
Visualization of Structure Matrix
Corner Detection Implementation

1. Compute image gradient
2. For each $m \times m$ neighborhood, compute matrix $C$ (optionally using weighted sum)
3. If smaller eigenvalue $\lambda_2$ is larger than threshold $\tau$, record a corner
4. Nonmaximum suppression: only keep strongest corner in each $m \times m$ window
Corner Detection Results

- Checkerboard with noise

Trucco & Verri
Corner Detection Results
Corner Detection Results

[Image of corner detection results]

Histogram of $\lambda_2$ (smaller eigenvalue)
Corner Detection

• Application: good features for tracking, correspondence, etc.
  – Why are corners better than edges for tracking?

• Other corner detectors
  – Look for maxima of curvature in edge detector output
  – Perform color segmentation on image, look for places where 3 segments meet
  – …
Invariance

- Suppose you rotate the image by some angle
  - Will you still find the same corners?

- What if you change the brightness?

- Scale?
Scale-Invariant Feature Detection

- Key idea: compute some function $f$ over different scales, find extremum
  - Common definition of $f$: convolution with LoG or DoG
  - Find local minima or maxima over position and scale
Blob Filter

- Recall: Laplacian of Gaussian
  - Circularly symmetric operator for blob detection

\[ \nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \]
Blob Detection – Single Scale

maxima

minima

maxima

Source: N. Snavely
Blob Detection – Over Multiple Scales

T. Lindeberg. Feature detection with automatic scale selection.
Multiscale Difference of Gaussians

Gaussian-filtered images with increasing $\sigma$

Difference-of-Gaussians Images
Automatic scale selection

Lindeberg et al., 1996
Automatic scale selection

Function responses for increasing scale

$f(I_{h..m}(x, \sigma))$
Automatic scale selection

Function responses for increasing scale

Scale-space programs

\[ f(I_{h\ldots m}(x,\sigma)) \]
Automatic scale selection

Function responses for increasing scale
Scale-space (example)

$f(L_{h_{i,m}}(x,\sigma))$
Automatic scale selection
Automatic scale selection
Automatic scale selection

Function responses for increasing scale (scale-space signature):

\[ f(I_{i \cdots i_m}(x, \sigma)) \]

\[ f(I_{i \cdots i_m}(x', \sigma')) \]
Automatic scale selection

Normalize: rescale to fixed size
Rotation Normalization

- Rotate window according to dominant orientation
  - Eigenvector of C corresponding to maximum eigenvalue
Detected Features

- Detected features with characteristic scales and orientations:

Feature Descriptors

- Once we have *detected* distinctive and repeatable features, still have to *match* them across images
  - Image alignment (e.g., mosaics), 3D reconstruction, motion tracking, object recognition, indexing and retrieval, robot navigation, etc.
Properties of Feature Descriptors

- Easily compared (compact, fixed-dimensional)
- Easily computed
- **Invariant**
  - Translation
  - Rotation
  - Scale
  - Change in image brightness
  - Change in perspective?
Scale Invariant Feature Transform

• Simple version:
  – Take 16×16 normalized window around detected feature
  – Create histogram of quantized gradient directions
  – Invariant to changes in brightness
Full SIFT Descriptor

- Divide 16×16 window into 4×4 grid of cells
- Compute an orientation histogram for each cell
  - 16 cells * 8 orientations = 128-dimensional descriptor

Properties of SIFT

- Fast (real-time) and robust descriptor for matching
  - Handles changes in viewpoint (≈60° out of plane rotation)
  - Handles significant changes in illumination
  - Lots of code available