Outline

• Concept of Scale
  – Pyramids
  – Scale-space approaches briefly
• Scale invariant region selection
• SIFT: an image region descriptor

• David Lowe, IJCV 2004 paper [Please read it!]
Reminder: Motivation

- Image Matching
  - Fundamental aspect of many problems

- Object Recognition
- 3D Structures
- Stereo Correspondence
- Motion Tracking
- ....
Reminder: Motivation

• Image Matching
  – Fundamental aspect of many problems

  • Object Recognition
  • 3D Structures
  • Stereo Correspondence
  • Motion Tracking
  • .....

What are the desired features to conduct these tasks?
Review of the Corner Detection

Corners are invariant to
- translation ✔
- rotation ✔
- scaling ✗
The extrema in a signal and its first a few derivatives provide a useful general purpose description for many kinds of signals

- Edges
- Corners
The extrema in a signal and its first a few derivatives provide a useful general purpose description for many kinds of signals

- Edges
- Corners

In physics, objects live on a range of scales.
The extrema in a signal and its first a few derivatives provide a useful general purpose description for many kinds of signals:

- Edges
- Corners

In physics, objects live on a range of scales.

Neighborhood operations can only extract local features (at a scale of at most a few pixels), however, images contain information at larger scales.
Scale-Space

• Any given image can contain objects that exist at scales different from other objects in the same image
Scale-Space

• Any given image can contain objects that exist at scales different from other objects in the same image

• Or, that even exist at multiple scales simultaneously
Scale-Space

• Any given image can contain objects that exist at scales different from other objects in the same image

• Or, that even exist at multiple scales simultaneously
Multi-scale Representation

Image

Small Scale Features → Smaller Filter Masks

Large Scale Features → Larger Filter Masks
Multi-scale Representation

Small Scale Features → Smaller Filter Masks
Large Scale Features → Larger Filter Masks

 Computational Cost increases!

Doubling the scale leads to four-fold increase in the number of operations in 2D.
Multi-Scale Representation: Pyramid

- **Pyramid** is one way to represent images in multi-scale
  - Pyramid is built by using multiple copies of image.
  - Each level in the pyramid is 1/4 of the size of previous level.
  - The lowest level is of the highest resolution.
  - The highest level is of the lowest resolution.
Multi-Scale Representation: Pyramid

Level 1: 1x1
Level 2: 2x2
Level 3: 4x4
Level 4: 8x8
Level 10: 512x512
Pyramid can capture global and local features
Gaussian Pyramids

512  256  128  64  32  16  8

Source: Forsyth
Laplacian Pyramids

512  256  128  64  32  16  8

Source: Forsyth
Laplacian of Gaussian (LoG): Gaussian first to smooth images then perform Laplacian operation

\[ \nabla^2 (G_\sigma \ast I) = I \ast \nabla^2 G_\sigma \]
Laplacian Pyramids
Laplacian Pyramids

\[
\frac{\delta G_\sigma}{\delta x}(x, y) = - \frac{x}{2\pi \sigma^4} e^{-(x^2 + y^2)/(2\sigma^2)}
\]
Laplacian Pyramids

\[ \frac{\delta G_\sigma}{\delta x}(x, y) = -\frac{x}{2\pi\sigma^4} e^{-\frac{(x^2 + y^2)}{(2\sigma^2)}} \]

Repeat same derivative for y component, and then take second derivatives.

\[ \nabla^2 G_\sigma(x, y) = \frac{1}{2\pi\sigma^4} \left( \frac{x^2 + y^2 - 2\sigma^2}{\sigma^2} \right) e^{-\frac{(x^2 + y^2)}{(2\sigma^2)}} \]
Laplacian Pyramid Construction
Laplacian Pyramid Construction

Diagram showing the construction process with "Interpolate" steps and images at different scales.
Decimation and Interpolation

Lowpass filter
(i.e., Gaussian)
Decimation and Interpolation

Lowpass filter (i.e., Gaussian)

\[ y(n) = x(n) \ast h(n) = \sum_{k} h(k)x(n - k) \]
Decimation and Interpolation

Lowpass filter (i.e., Gaussian)

\[ y(n) = x(n) \ast h(n) = \sum_k h(k)x(n - k) \]

\[ z(n) = y(2n) \]
Decimation and Interpolation

Lowpass filter (i.e., Gaussian)

\[ y(n) = x(n) \ast h(n) = \sum_k h(k)x(n - k) \]

\[ z(n) = y(2n) \]

\[ z(n) = \sum_k h(k)x(2n - k) \]
Difference of Gaussian (DoG)

- It is a common approximation of LoG (better run time).
Difference of Gaussian (DoG)

• It is a common approximation of LoG (better run time).

\[ D_{\sigma, \alpha}(x, y) = L(x, y, \alpha) - L(x, y, \alpha \sigma) \]
Difference of Gaussian (DoG)

• It is a common approximation of LoG (better run time).

\[ D_{\sigma, \alpha}(x, y) = L(x, y, \alpha) - L(x, y, \alpha \sigma) \]

• It is the difference between a blurred copy of image I and an even more blurred copy of I.
Difference of Gaussian (DoG)

- It is a common approximation of LoG (better run time).

\[ D_{\sigma, \alpha}(x, y) = L(x, y, \alpha) - L(x, y, \alpha \sigma) \]

- It is the difference between a blurred copy of image I and an even more blurred copy of I.

\[ \nabla G_{\sigma}(x, y) \approx \frac{G_{\alpha \sigma}(x, y) - G_{\sigma}(x, y)}{(\alpha - 1)\sigma^2} \]
Scale Selection-Automated

- Find scale that gives local maxima of some function $f$ in both position and scale.

K. Grauman,
Good Function?

• A “good” function for scale detection: has one stable sharp peak

• For usual images: a good function would be a one which responds to contrast (sharp local intensity change)
Patch Size Corresponding to Scale

How to find corresponding patch sizes?

\[ f(I_{i_1 \ldots i_m}(x, \sigma)) = f(I_{i_1 \ldots i_m}(x', \sigma')) \]
Patch Size Corresponding to Scale

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

\[ f(I_{i_1...i_m}(x', \sigma)) \]
Patch Size Corresponding to Scale

\[ f(I_{i_1 \ldots i_m} (x, \sigma)) \]

\[ f(I_{i_1 \ldots i_m} (x', \sigma)) \]
Patch Size Corresponding to Scale

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

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

K. Grauman, B. Leibe
Patch Size Corresponding to Scale

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

\[ f(I_{i_1...i_m}(x', \sigma)) \]
Patch Size Corresponding to Scale

\[ f(I_{i_1 \ldots i_m}(x, \sigma)) \]

\[ f(I_{i_1 \ldots i_m}(x', \sigma)) \]
Patch Size Corresponding to Scale

- Function responses for increasing scale (scale signature)
Normalization

Normalize: rescale to fixed size
Scale Invariant Feature Transform (SIFT)

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

• Shows existence, importance, and value of invariant types of detector
• Demonstrates the richness that feature descriptors can bring to feature matching

Instead of using LoG (Laplacian of Gaussian) like in Harris and Hessian-based operators, SIFT uses DoG (Difference of Gaussian).
Scale Invariant Feature Transform (SIFT)

- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters
Scale Invariant Feature Transform (SIFT)

- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters.
Overall Procedure at a 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 Descriptor

Basic idea:
- Take n x n (i.e., n=16) square window (around a *feature/interest point*)
- Divide them into m x m (i.e, m=4) cells
- Compute gradient orientation for each cell
- Create histogram over edge orientations weighted by magnitude

![Image gradients]

![angle histogram]
SIFT Descriptor

16 histograms \( \times \) 8 orientations
\[ = 128 \] features
**SIFT Descriptor**

**Full version**
- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor
- Threshold normalize the descriptor:

\[ \sum_{i} d_{i}^{2} = 1 \quad \text{such that:} \quad d_{i} < 0.2 \]
Properties of **SIFT**

Extraordinarily robust matching technique

- Can handle changes in viewpoint
- Can handle significant changes in illumination
- Efficient (real time)
- Lots of code available
Revisit SIFT Steps

(1) Scale-space extrema detection
   – Extract scale and rotation invariant interest points (i.e., keypoints).

(2) Keypoint localization
   – Determine location and scale for each interest point.
   – Eliminate “weak” keypoints

(3) Orientation assignment
   – Assign one or more orientations to each keypoint.

(4) Keypoint descriptor
   – Use local image gradients at the selected scale.
(1) Scale-Space Extrema Detection

- We want to find points that give us information about the objects in the image.

- The information about the objects is in the object’s edges.

- We will represent the image in a way that gives us these edges as this representations extrema points.
(1) Scale-Space Extrema Detection

- **Harris-Laplace**
  
  *Find local maxima of:*
  - Harris detector in space
  - LoG in scale

- **SIFT**
  
  *Find local maxima of:*
  - Hessian in space
  - DoG in scale
(1) Scale-Space Extrema Detection
(1) Scale-Space Extrema Detection

- Extract local extrema (i.e., minima or maxima) in DoG pyramid.
  - Compare each point to its 8 neighbors at the same level, 9 neighbors in the level above, and 9 neighbors in the level below (i.e., 26 total).
(1) Scale-Space Extrema Detection

- **Laplacian of Gaussian** kernel
  - Scale normalized (x by scale$^2$)
  - Proposed by Lindeberg

- **Scale-space detection**
  - Find local maxima across scale/space
  - A good “blob” detector

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

\[
\nabla^2 G(x, y, \sigma) = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2}
\]
(2) Keypoint Localization

- There are still a lot of points, some of them are not good enough.
- The locations of keypoints may be not accurate.
- Eliminating edge points.
(2) Keypoint Localization

- Reject (1) points with low contrast (flat)
  (2) poorly localized along an edge (edge)
(2) Keypoint Localization

- Reject (1) points with low contrast (flat)
- (2) poorly localized along an edge (edge)

\[
\text{if } \frac{Tr(H)}{|H|} < \frac{(r + 1)^2}{r}
\]

Reject

Where \( r = \frac{\lambda_1}{\lambda_2} \) and practically SIFT uses \( r = 10 \).
Inaccurate Keypoint Localization

- Poor contrast
Inaccurate Keypoint Localization

The Solution:

- Taylor expansion:

\[ D(\tilde{x}) = D + \frac{\partial D^T}{\partial \tilde{x}} \tilde{x} + \frac{1}{2} \tilde{x}^T \frac{\partial^2 D^T}{\partial \tilde{x}^2} \tilde{x} \]

- Minimize to find accurate extrema:

\[ \hat{x} = -\frac{\partial^2 D}{\partial \tilde{x}^2}^{-1} \frac{\partial D}{\partial \tilde{x}} \]

- If offset from sampling point is larger than 0.5 - Keypoint should be in a different sampling point.

Brown & Lowe 2002
Local extremas
Local extremas

Remove low contrast features
Local extremas

Remove low edges
SIFT Descriptor
(3) Orientation Assignment

Create histogram of gradient directions, within a region around the keypoint, at selected scale:

\[ L(x, y, \sigma) = G(x, y, \sigma) \times I(x, y) \]

\[ m(x, y) = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2} \]

\[ \theta(x, y) = \arctan\frac{(L(x, y+1) - L(x, y-1))}{(L(x+1, y) - L(x-1, y))} \]

36 bins (i.e., 10° per bin)

- Histogram entries are weighted by (i) gradient magnitude and (ii) a Gaussian function with \( \sigma \) equal to 1.5 times the scale of the keypoint.
(3) Orientation Assignment

\[ \text{Dx} \quad \text{Dy} \]

\[ \text{M} \quad \Theta \]
(3) Orientation Assignment

- Keypoint location = extrema location
- Keypoint scale is scale of the DOG image
(3) Orientation Assignment

gradient magnitude  weighted by 2D gaussian kernel  weighted gradient magnitude
(3) Orientation Assignment

weighted gradient magnitude

weighted orientation histogram.
Each bucket contains sum of weighted gradient magnitudes corresponding to angles that fall within that bucket.

36 buckets
10 degree range of angles in each bucket, i.e.
  0 <= ang < 10 : bucket 1
  10 <= ang < 20 : bucket 2
  20 <= ang < 30 : bucket 3 ...
(3) Orientation Assignment

Orientation of keypoint is approximately 25 degrees.
(4) Keypoint Descriptor

- **Partial Voting**: distribute histogram entries into adjacent bins (i.e., **additional robustness to shifts**)
  - Each entry is added to all bins, multiplied by a weight of \(1-d\), where \(d\) is the distance from the bin it belongs.
(4) Keypoint Descriptor

- Descriptor depends on two main parameters:
  - (1) number of orientations
  - n x n array of orientation histograms
Evaluating Results

How can we measure the performance of a feature matcher?

- **True positive rate**:
  \[
  \frac{\text{# true positives matched}}{\text{# true positives}}
  \]
- **False positive rate**:
  \[
  \frac{\text{# false positives matched}}{\text{# true negatives}}
  \]

- The matcher correctly found a match:
  - The matcher said yes when the right answer was yes.
  - The matcher correctly found a match.
- Features that really do have a match:
  - Features that really do have a match.

Graph:
- X-axis: False positive rate
- Y-axis: True positive rate
- Point (0.1, 0.7) on the graph represents a false positive rate of 0.1 and a true positive rate of 0.7.
Evaluating Results

How can we measure the performance of a feature matcher?

ROC curve ("Receiver Operator Characteristic")

- True positive rate: \[ \frac{\text{# true positives matched}}{\text{# true positives}} \]
- False positive rate: \[ \frac{\text{# false positives matched}}{\text{# true negatives}} \]

Features that really do have a match:
- The matcher correctly found a match

Features that do not have a match:
- The matcher said yes when the right answer was no
Change of Illumination

- **Change of brightness** => doesn’t effect gradients (difference of pixels value).
- **Change of contrast** => doesn’t effect gradients (up to normalization).
- **Saturation** (non-linear change of illumination) => affects magnitudes much more than orientation.
  => Threshold gradient magnitudes to 0.2 and renormalize.
1. Before you start, you can normalize the images.
2. Difference based metrics can be used (Haar, SIFT,...)
• Extract features
• Extract features
• Compute *putative matches*
• Extract features

• Compute *putative matches*

• Loop:
  – *Hypothesize* transformation $T$ (small group of putative matches that are related by $T$)
• Extract features
• Compute *putative matches*
• Loop:
  – *Hypothesize* transformation $T$ (small group of putative matches that are related by $T$)
  – *Verify* transformation (search for other matches consistent with $T$)
• Extract features
• Compute *putative matches*
• Loop:
  – *Hypothesize* transformation $T$ (small group of putative matches that are related by $T$)
  – *Verify* transformation (search for other matches consistent with $T$)
Object Recognition

- **For training images:**
  - Extracting keypoints by SIFT.
  - Creating descriptors database.

- **For query images:**
  - Extracting keypoints by SIFT.
  - For each descriptor - finding nearest neighbor in DB.
  - Finding cluster of at-least 3 keypoints.
  - Performing detailed geometric fit check for each cluster.
Recognition/Matching
Image Registration