Lecture 18: Scale Invariant Features (SIFT)
Menu for Today (February 26, 2021)

Topics:

— Scale Invariant Feature Transform (SIFT)
— SIFT Detector
— SIFT Descriptor
— Analysis of stability

Readings:

— Today’s Lecture: Forsyth & Ponce (2nd ed.) 5.4
  “Distinctive Image Features for Scale-Invariant Keypoints
— Next Lecture: Forsyth & Ponce (2nd ed.) 10.4.2, 10.1, 10.2

Reminders:

— Assignment 3: Texture Synthesis is out, due on Mar 1
Today’s “fun” Example: Recognizing Panoramas

Figure Credit: Matthew Brown and David Lowe
Figure Credit: Matthew Brown and David Lowe
Figure Credit: Matthew Brown and David Lowe
Figure Credit: Matthew Brown and David Lowe
Figure Credit: Matthew Brown and David Lowe
Figure Credit: Matthew Brown and David Lowe
Figure Credit: Matthew Brown and David Lowe
Figure Credit: Matthew Brown and David Lowe
Example 1: Color Matching Experiment

Example Credit: Bill Freeman
Example 1: Color Matching Experiment

\[ T = w_1 P_1 + w_2 P_2 + w_3 P_3 \]
**Example 2: Color Matching Experiment**

We say a “negative” amount of $P_2$ was needed to make a match because we added it to the test color side.

The primary color amount needed to match:

\[ p_1 \quad p_2 \quad p_3 \]
Color Matching Experiments

— positive weighted sum of A, B, C

\[ M = aA + bB + cC \]

— This is **additive** matching (a, b, c)
— Some colours can’t be matched this way

— Instead, we must write

where the = sign should be read as “matches”

\[ M + aA = bB + cC \]

— This is **subtractive** matching

— Interpret this as (−a, b, c)

Problem for **designing displays**: Choose phosphors R, G, B so that **positive linear combinations** match a large set of colours
Color Matching Experiments

— Some colours can’t be matched this way

— Instead, we must write

\[ M + aA = bB + cC \]

where, again, the = sign should be read as “matches”

— This is **subtractive** matching

— Interpret this as (−a, b, c)
Human **Cone** Sensitivity

http://hyperphysics.phy-astr.gsu.edu/hbase/vision/colcon.html
Linear Color Spaces

A choice of primaries yields a linear colour space
— the coordinates of a colour are given by the weights of the primaries used to match it

Choice of primaries is equivalent to choice of colour space

— **RGB**: Primaries are monochromatic energies, say 645.2 nm, 526.3 nm, 444.4 nm

— **CIE XYZ**: Primaries are imaginary, but have other convenient properties. Colour coordinates are \((X, Y, Z)\), where \(X\) is the amount of the \(X\) primary, etc.
Geometry of Colour (CIE)

- White is in the center, with saturation increasing towards the boundary.
- Mixing two coloured lights creates colours on a straight line.
- Mixing 3 colours creates colours within a triangle.
- Curved edge means there are no 3 actual lights that can create all colours that humans perceive!
Geometry of Colour (CIE)

- White is in the center, with saturation increasing towards the boundary.
- Mixing two coloured lights creates colours on a straight line.
- Mixing 3 colours creates colours within a triangle.
- Curved edge means there are no 3 actual lights that can create all colours that humans perceive!
RGB Colour Space

The subspace of CIE colours that can be displayed on a typical computer monitor (phosphor limitations keep the space quite small)
HSV Colour Space

More natural description of colour for human interpretation

**Hue**: attribute that describes a pure colour
— e.g. ’red’, ’blue’

**Saturation**: measure of the degree to which a pure colour is diluted by white light
— pure spectrum colours are fully saturated

**Value**: intensity or brightness

Hue + saturation also referred to as **chromaticity**.
HSV Colour Space

Gonzalez and Woods, 2008
Colour **Constancy**

Image colour depends on both light colour and surface colour

**Colour constancy**: determine hue and saturation under different colours of lighting

It is surprisingly difficult to predict what colours a human will perceive in a complex scene
— depends on context, other scene information

Humans can usually perceive
— the colour a surface would have under white light
Colour Constancy

A classic experiment by Land and McCann
Environmental Effects

**Chromatic adaptation**: If the human visual system is exposed to a certain colour light for a while, colour perception starts to skew

**Contrast effects**: Nearby colours affect what is perceived
Additive and subtractive colour

Additive color: Light
RGB primaries; CRT monitors

Subtractive color: Ink
CMY primaries; Film, prints

From megagrafxstudio
Summary

— Human colour perception
  — colour matching experiments
  — additive and subtractive matching
  — principle of trichromacy

— RGB and CIE XYZ are linear colour spaces

— Uniform colour space: differences in coordinates are a good guide to differences in perceived colour

— HSV colour space: more intuitive description of colour for human interpretation

— (Human) colour constancy: perception of intrinsic surface colour under different colours of lighting
Where are the good features, and how do we match them?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Photometric Transformations
Geometric Transformations

objects will appear at different scales, translation and rotation

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Lets assume for the moment we can figure out where the good features (patches) are ... how do we **match** them?
How do we know which corner goes with which?
How do we know which blob goes with which?
Patch around the local feature is very informative
Intensity Image

Just use the pixel values of the patch

Perfectly fine if geometry and appearance is unchanged (a.k.a. template matching)

What are the problems?

vector of intensity values

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Just use the pixel values of the patch

Perfectly fine if geometry and appearance is unchanged (a.k.a. template matching)

What are the problems?

How can you be less sensitive to absolute intensity values?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Image **Gradients / Edges**

Use pixel differences

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
</tbody>
</table>

(− + + − − − +)

vector of x derivatives

Feature is invariant to absolute intensity values

What are the problems?
Use pixel differences

\[
\begin{array}{ccc}
1 & 2 & 3 \\
4 & 5 & 6 \\
7 & 8 & 9 \\
\end{array}
\rightarrow \begin{pmatrix}
- & + & + & - & - & + \\
\end{pmatrix}
\]

vector of x derivatives

Feature is invariant to absolute intensity values

What are the problems?

How can you be less sensitive to deformations?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Where does **SIFT** fit in?

<table>
<thead>
<tr>
<th>Representation</th>
<th>Result is...</th>
<th>Approach</th>
<th>Technique</th>
</tr>
</thead>
<tbody>
<tr>
<td>intensity</td>
<td>dense (2D)</td>
<td>template matching</td>
<td>(normalized) correlation, SSD</td>
</tr>
<tr>
<td>edge</td>
<td>relatively sparse (1D)</td>
<td>derivatives</td>
<td>$\nabla^2 G$, Canny</td>
</tr>
<tr>
<td>“corner”</td>
<td>sparse (0D)</td>
<td>locally distinct features</td>
<td>Harris, SIFT</td>
</tr>
</tbody>
</table>
Object Recognition with Invariant Features

**Task:** Identify objects or scenes and determine their pose and model parameters

**Applications:**

— Industrial automation and inspection
— Mobile robots, toys, user interfaces
— Location recognition
— Digital camera panoramas
— 3D scene modeling, augmented reality
David Lowe’s Invariant Local Features

Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters.
David Lowe’s Invariant Local Features

Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters.
Advantages of Invariant Local Features

**Locality**: features are local, so robust to occlusion and clutter (no prior segmentation)

**Distinctiveness**: individual features can be matched to a large database of objects

**Quantity**: many features can be generated for even small objects

**Efficiency**: close to real-time performance
Scale Invariant Feature Transform (SIFT)

SIFT describes both a detector and descriptor

1. Multi-scale extrema detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
1. Multi-scale Extrema Detection

- First octave
- Second octave

Gaussian
Difference of Gaussian (DoG)

Half the size

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Recall: Applying Laplacian Filter at Different Scales

Slide Credit: Ioannis (Yannis) Gkioulakas (CMU)
Recall: Template matching

Image Pyramid(s)  Template Pyramid (1/s)

Both allow search over scale
Recall: Applying **Laplacian** Filter at Different **Scales**

Full size

3/4 size

*Slide Credit:* Ioannis (Yannis) Gkioulekas (CMU)
Searching over \textbf{Scale}-space
Searching over \textbf{Scale}-space

\[ \sigma \]

\[ \sigma' = 2\sigma \]

\[ \sigma' = 3\sigma \]
Searching over **Scale-space**

\[ \sigma \]

\[ \sigma \]

\[ \sigma \]

\[ s = 0.5 \]

\[ s = 0.33 \]
1. Multi-scale Extrema Detection

Gaussian

Laplacian
1. Multi-scale Extrema Detection

Detect maxima and minima of Difference of Gaussian in scale space

\[ \sigma = 2^{1/s} \]

Selected if larger than all 26 neighbors

Difference of Gaussian (DoG)

Slide Credit: Ioannis (Yannis) Gkioullekas (CMU)
1. Multi-scale Extrema Detection — Sampling Frequency

More points are found as sampling frequency increases, but accuracy of matching decreases after 3 scales/octave

![Graph showing the relationship between number of scales sampled per octave and correctly matched keypoints.]
2. Keypoint Localization

— After keypoints are detected, we remove those that have **low contrast** or are **poorly localized** along an edge
2. Keypoint Localization

— After keypoints are detected, we remove those that have **low contrast** or are **poorly localized** along an edge.

How do we decide whether a keypoint is poorly localized, say along an edge, vs. well-localized?
2. Keypoint Localization

— After keypoints are detected, we remove those that have low contrast or are poorly localized along an edge.

How do we decide whether a keypoint is poorly localized, say along an edge, vs. well-localized?

\[
C = \begin{bmatrix}
\sum_{p \in P} I_x I_x & \sum_{p \in P} I_x I_y \\
\sum_{p \in P} I_y I_x & \sum_{p \in P} I_y I_y
\end{bmatrix}
\]
2. Keypoint Localization

— After keypoints are detected, we remove those that have **low contrast** or are **poorly localized** along an edge

How do we decide whether a keypoint is poorly localized, say along an edge, vs. well-localized?

— Lowe suggests computing the ratio of the eigenvalues of $C$ (recall Harris corners) and checking if it is greater than a threshold

— Aside: The ratio can be computed efficiently in fewer than 20 floating point operations, using a trick involving the trace and determinant of $C$ - no need to explicitly compute the eigenvalues
2. Keypoint Localization

Example:

(a) $233 \times 189$ image
(b) 832 DOG extrema
(c) 729 left after peak value threshold
(d) 536 left after testing ratio of principal curvatures
3. Orientation Assignment

— Create **histogram** of local gradient directions computed at selected scale

— Assign **canonical orientation** at peak of smoothed histogram

— Each key specifies stable 2D coordinates \((x, y, \text{scale, orientation})\)
3. Orientation Assignment

| ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← |

Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length)
### 3. Orientation Assignment

| ← ← → → → → → → |← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → |
| ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → |
| ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → |
| ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → |
| ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → |
| ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → |
| ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → |
| ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → |
| ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → |
| ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → | ← ← → → → → → → |

Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length).
3. Orientation Assignment

Arrows illustrate gradient orientation (direction) and gradient magnitude (arrow length)
3. Orientation Assignment

Arrows illustrate gradient orientation (direction) and gradient magnitude (arrow length).
3. Orientation Assignment

Arrows illustrate gradient orientation (direction) and gradient magnitude (arrow length).
3. Orientation Assignment

Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length).
### 3. Orientation Assignment

Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length).

| ← | ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← | ← |
| ← | ← | ← | ← | ← | ← | ← | ← | ← |

Assigned Orientation
3. Orientation Assignment

Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length).
3. Orientation Assignment

Multiply gradient magnitude by a Gaussian kernel

| ← | ← | ● | ● | ● | ● | ● | ● | ● |
| ← | ← | ● | ● | ● | ● | ● | ● | ● |
| ← | ← | ● | ● | ● | ● | ● | ● | ● |
| ← | ← | ● | ● | ● | ● | ● | ● | ● |
| ← | ← | ● | ● | ● | ● | ● | ● | ● |
| ← | ← | ● | ● | ● | ● | ● | ● | ● |
| ← | ← | ● | ● | ● | ● | ● | ● | ● |
| ← | ← | ● | ● | ● | ● | ● | ● | ● |

Arrows illustrate gradient orientation (direction) and gradient magnitude (arrow length)
3. Orientation Assignment

- **Histogram** of 36 bins (10 degree increments)

- Size of the **window** is 1.5 scale (recall the Gaussian filter)

- Gaussian-weighted **voting**

- Highest **peak** and peaks above 80% of highest also considered for calculating dominant orientations
4. Keypoint Description

We have seen how to assign a location, scale, and orientation to each key point—**keypoint detection**

— The next step is to compute a **keypoint descriptor**: should be robust to local shape distortions, changes in illumination or 3D viewpoint

— Keypoint detection is not the same as keypoint description, e.g. some applications skip keypoint detection and extract SIFT descriptors on a regularly spaced grid
4. SIFT Descriptor

— Thresholded image gradients are sampled over $16 \times 16$ array of locations in scale space (weighted by a Gaussian with sigma half the size of the window)

— Create array of orientation histograms

— 8 orientations $\times 4 \times 4$ histogram array
How many dimensions are there in a SIFT descriptor?

(Hint: This diagram shows a 2 x 2 histogram array but the actual descriptor uses a 4 x 4 histogram array)