Image gradients and edges

Tues Jan 24, 2017
Kristen Grauman
UT-Austin

Announcements

• Slides are posted for lecture the night before
• Office hours on homepage
  – Tues 11-12 + appointment (me)
  – Tues 3-4 and Wed 4-5 (Nick)
  – Mon 2:30-3:30 and Thurs 3:30-4:30 (Paul)
• Reminder: no laptops, phones, tablets, etc. open in class.
• Class is 100% full with registered students. Please reserve chairs for those on the roster.
Last time

- Various models for image “noise”
- Linear filters and convolution useful for
  - Image smoothing, removing noise
    - Box filter
    - Gaussian filter
    - Impact of scale / width of smoothing filter
- Separable filters more efficient
- Median filter: a non-linear filter, edge-preserving

Image filtering

- Compute a function of the local neighborhood at each pixel in the image
  - Function specified by a “filter” or mask saying how to combine values from neighbors.

- Uses of filtering:
  - Enhance an image (denoise, resize, etc)
  - Extract information (texture, edges, etc)
  - Detect patterns (template matching)

Adapted from Derek Hoiem
Edge detection

- **Goal**: map image from 2d array of pixels to a set of curves or line segments or contours.
- **Why**?

  - **Main idea**: look for strong gradients, post-process

What causes an edge?

- Reflectance change: appearance information, texture
- Depth discontinuity: object boundary
- Cast shadows
- Change in surface orientation: shape
Edges/gradients and invariance

Derivatives and edges

An edge is a place of rapid change in the image intensity function.

Source: L. Lazebnik
Derivatives with convolution

For 2D function, \( f(x,y) \), the partial derivative is:

\[
\frac{\partial f(x,y)}{\partial x} = \lim_{\varepsilon \to 0} \frac{f(x + \varepsilon, y) - f(x, y)}{\varepsilon}
\]

For discrete data, we can approximate using finite differences:

\[
\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x+1,y) - f(x,y)}{1}
\]

To implement above as convolution, what would be the associated filter?

Partial derivatives of an image

\[
\frac{\partial f(x,y)}{\partial x}
\]

\[
\frac{\partial f(x,y)}{\partial y}
\]

Which shows changes with respect to \( x \)?

(showing filters for correlation)
**Image gradient**

The gradient of an image:

\[ \nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right] \]

The gradient points in the direction of most rapid change in intensity.

- \( \nabla f = \left[ \frac{\partial f}{\partial x}, 0 \right] \)
- \( \nabla f = \left[ 0, \frac{\partial f}{\partial y} \right] \)
- \( \nabla f = [0, 0] \)

The **gradient direction** (orientation of edge normal) is given by:

\[ \theta = \tan^{-1} \left( \frac{\partial f}{\partial y} / \frac{\partial f}{\partial x} \right) \]

---

**Effects of noise**

Consider a single row or column of the image:

- Plotting intensity as a function of position gives a signal.

Where is the edge?
Solution: smooth first

Where is the edge? Look for peaks in $\frac{\partial}{\partial x}(h \ast f)$

Derivative theorem of convolution

$\frac{\partial}{\partial x}(h \ast f) = (\frac{\partial}{\partial x}h) \ast f$

Differentiation property of convolution.
Derivative of Gaussian filters

\[(I \otimes g) \otimes h = I \otimes (g \otimes h)\]

\[
\begin{bmatrix}
0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\
0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\
0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\
0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\
0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030
\end{bmatrix} \otimes \begin{bmatrix} 1 & -1 \end{bmatrix}
\]

Derivative of Gaussian filters

Source: L. Lazebnik
Laplacian of Gaussian

Consider \( \frac{\partial^2}{\partial x^2} (h * f) \)

\[
\begin{align*}
\text{Gaussian} & \quad h_\sigma(u, v) = \frac{1}{2\pi \sigma^2} e^{-\frac{u^2 + v^2}{2\sigma^2}} \\
\text{derivative of Gaussian} & \quad \frac{\partial}{\partial x} h_\sigma(u, v) \\
\text{Laplacian of Gaussian} & \quad \nabla^2 h_\sigma(u, v)
\end{align*}
\]

Where is the edge? Zero-crossings of bottom graph

Slide credit: Steve Seitz

2D edge detection filters

- \( \nabla^2 \) is the Laplacian operator:
  \[
  \nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}
  \]

Slide credit: Steve Seitz
Smoothing with a Gaussian

Recall: parameter $\sigma$ is the “scale” / “width” / “spread” of the Gaussian kernel, and controls the amount of smoothing.

Effect of $\sigma$ on derivatives

The apparent structures differ depending on Gaussian’s scale parameter.

Larger values: larger scale edges detected
Smaller values: finer features detected
So, what scale to choose?

It depends what we’re looking for.

Mask properties

• **Smoothing**
  – Values positive
  – Sum to 1 → constant regions same as input
  – Amount of smoothing proportional to mask size
  – Remove “high-frequency” components; “low-pass” filter

• **Derivatives**
  – _________ signs used to get high response in regions of high contrast
  – Sum to ___ → no response in constant regions
  – High absolute value at points of high contrast
Seam carving: main idea

[Shai & Avidan, SIGGRAPH 2007]
Seam carving: main idea

Real image example
Seam carving: main idea

Intuition:

- Preserve the most “interesting” content
  → Prefer to remove pixels with low gradient energy
- To reduce or increase size in one dimension, remove irregularly shaped “seams”
  → Optimal solution via dynamic programming.

Seam carving: main idea

\[ \text{Energy}(f) = \sqrt{\left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2} \]

- Want to remove seams where they won’t be very noticeable:
  – Measure “energy” as gradient magnitude
- Choose seam based on minimum total energy path across image, subject to 8-connectedness.
Let a **vertical seam** \( s \) consist of \( h \) positions that form an 8-connected path.

Let the **cost of a seam** be: \( \text{Cost}(s) = \sum_{i=1}^{h} \text{Energy}(f(s_i)) \)

**Optimal seam** minimizes this cost: \( s^* = \min_s \text{Cost}(s) \)

Compute it efficiently with **dynamic programming**.

---

**How to identify the minimum cost seam?**

- First, consider a **greedy** approach:

\[
\begin{array}{ccc}
1 & 3 & 0 \\
2 & 8 & 9 \\
5 & 2 & 6 \\
\end{array}
\]

**Energy matrix**

(gradient magnitude)

\[
\text{Energy}(f) = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}
\]
Seam carving: algorithm

- Compute the cumulative minimum energy for all possible connected seams at each entry \((i,j)\):
  \[ M(i, j) = Energy(i, j) + \min(M(i-1, j-1), M(i-1, j), M(i-1, j+1)) \]

- Then, min value in last row of \(M\) indicates end of the minimal connected vertical seam.
- Backtrack up from there, selecting min of 3 above in \(M\).

\[
\begin{array}{cccc}
 1 & 3 & 0 \\
 2 & 8 & 9 \\
 5 & 2 & 6 \\
\end{array}
\]

Example

\[ M(i, j) = Energy(i, j) + \min(M(i-1, j-1), M(i-1, j), M(i-1, j+1)) \]
Example

\[ M(i, j) = \text{Energy}(i, j) + \min(M(i-1, j-1), M(i-1, j), M(i-1, j+1)) \]

Original Image

Energy Map

Blue = low energy
Red = high energy
Real image example

Other notes on seam carving

- Analogous procedure for horizontal seams
- Can also insert seams to increase size of image in either dimension
  - Duplicate optimal seam, averaged with neighbors
- Other energy functions may be plugged in
  - E.g., color-based, interactive,…
- Can use combination of vertical and horizontal seams
Example results from prior classes

(a) Original input

(b) Content-aware resizing

(c) Image from ‘imresize’

Results from Eunho Yang

Original image

Conventional resize

Seam carving result

Results from Martin Becker
Results from Martin Becker

Results from Jay Hennig
Removal of a marked object

(a) Selected an area.

Results from Eunho Yang
“Failure cases” with seam carving

By Donghyuk Shin

“Failure cases” with seam carving

By Suyog Jain
Gradients -> edges

Primary edge detection steps:
1. Smoothing: suppress noise
2. Edge enhancement: filter for contrast
3. Edge localization
   - Determine which local maxima from filter output are actually edges vs. noise
     • Threshold, Thin

Thresholding

• Choose a threshold value t
• Set any pixels less than t to zero (off)
• Set any pixels greater than or equal to t to one (on)
Original image

Gradient magnitude image
Thresholding gradient with a lower threshold

Thresholding gradient with a higher threshold
Canny edge detector

- Filter image with derivative of Gaussian
- Find magnitude and orientation of gradient
- **Non-maximum suppression:**
  - Thin wide “ridges” down to single pixel width
- **Linking and thresholding (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');`
- `>>help edge`

Source: D. Lowe, L. Fei-Fei

---

The Canny edge detector

original image (Lena)

Slide credit: Steve Seitz
The Canny edge detector

norm of the gradient

The Canny edge detector

thresholding
The Canny edge detector

How to turn these thick regions of the gradient into curves?

Non-maximum suppression

Check if pixel is local maximum along gradient direction, select single max across width of the edge
  • requires checking interpolated pixels p and r
The Canny edge detector

Problem: pixels along this edge didn’t survive the thresholding

thinning
(non-maximum suppression)

Hysteresis thresholding

- Threshold at low/high levels to get weak/strong edge pixels
- Do connected components, starting from strong edge pixels
Hysteresis thresholding

• Use a high threshold to start edge curves, and a low threshold to continue them.

Source: Steve Seitz

Final Canny Edges

Credit: James Hays
Recap: Canny edge detector

- Filter image with derivative of Gaussian
- Find magnitude and orientation of gradient
- **Non-maximum suppression:**
  - Thin wide “ridges” down to single pixel width
- **Linking and thresholding (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’);`
- `>>help edge`

Source: D. Lowe, L. Fei-Fei

Low-level edges vs. perceived contours

Background  Texture  Shadows
## Low-level edges vs. perceived contours

<table>
<thead>
<tr>
<th>Image</th>
<th>Human Segmentation</th>
<th>Gradient Magnitude</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Image" /></td>
<td><img src="human_segmentation" alt="Segmentation" /></td>
<td><img src="gradient" alt="Gradient" /></td>
</tr>
</tbody>
</table>

Berkeley segmentation database:  
[http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/](http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/)

Source: L. Lazebnik

---

## Protocol

*You will be presented a photographic image. Divide the image into some number of segments, where the segments represent “things” or “parts of things” in the scene. The number of segments is up to you, as it depends on the image. Something between 2 and 30 is likely to be appropriate. It is important that all of the segments have approximately equal importance.*

- Custom segmentation tool
- Subjects obtained from work-study program (UC Berkeley undergraduates)

Berkeley Segmentation Data Set  
David Martin, Charless Fowlkes, Doron Tal, Jitendra Malik  
Credit: David Martin
Learn from humans which combination of features is most indicative of a "good" contour?

[D. Martin et al. PAMI 2004]  Human-marked segment boundaries

Dataflow

Image → Boundary Cues: Brightness, Color, Texture → Cue Combination → Model → $P_b$

Challenges: texture cue, cue combination
Goal: learn the posterior probability of a boundary $P_b(x,y,0)$ from local information only
What features are responsible for perceived edges?

Feature profiles (oriented energy, brightness, color, and texture gradients) along the patch’s horizontal diameter.

[D. Martin et al. PAMI 2004] Kristen Grauman, UT-Austin
Brightness and Color Features

- 1976 CIE L*a*b* colorspace
- Brightness Gradient $\text{BG}(x,y,r,\theta)$
  - $\chi^2$ difference in $L^*$ distribution
- Color Gradient $\text{CG}(x,y,r,\theta)$
  - $\chi^2$ difference in $a^*$ and $b^*$ distributions

$$\chi^2(g,h) = \frac{1}{2} \sum_i \frac{(g_i - h_i)^2}{g_i + h_i}$$

---

Dataflow

- **Image**
- Optimized Cues:
  - Brightness
  - Color
  - Texture
- Human Segmentations
- **P_b**
- **Benchmark**
  - Model
  - Cue Combination

Credit: David Martin
Contour Detection

Canny+opt thresholds
Canny
Prewitt, Sobel, Roberts

Human agreement
Learned with combined features

UC Berkeley
Source: Jitendra Malik:
http://www.cs.berkeley.edu/~malik/malik-talks.ptrs.html

Computer Vision Group

[D. Martin et al. PAMI 2004] Kristen Grauman, UT Austin
Recall: image filtering

• Compute a function of the local neighborhood at each pixel in the image
  – Function specified by a “filter” or mask saying how to combine values from neighbors.

• Uses of filtering:
  – Enhance an image (denoise, resize, etc)
  – Extract information (texture, edges, etc)
  – Detect patterns (template matching)

Filters for features

• Map raw pixels to an intermediate representation that will be used for subsequent processing

• Goal: reduce amount of data, discard redundancy, preserve what’s useful
Template matching

• Filters as templates:
  Note that filters look like the effects they are intended to find --- “matched filters”

• Use normalized cross-correlation score to find a given pattern (template) in the image.
• Normalization needed to control for relative brightnesses.

A toy example
Template matching

Detected template

Template

Template matching

Detected template

Correlation map
Where’s Waldo?

Template

Scene

Where’s Waldo?

Detected template

Template
Where’s Waldo?

Detected template

Correlation map

Template matching

What if the template is not identical to some subimage in the scene?
Template matching

Match can be meaningful, if scale, orientation, and general appearance is right.
How to find at any scale?

Recap: Mask properties

- **Smoothing**
  - Values positive
  - Sum to 1 → constant regions same as input
  - Amount of smoothing proportional to mask size
  - Remove “high-frequency” components; “low-pass” filter

- **Derivatives**
  - Opposite signs used to get high response in regions of high contrast
  - Sum to 0 → no response in constant regions
  - High absolute value at points of high contrast

- **Filters act as templates**
  - Highest response for regions that “look the most like the filter”
  - Dot product as correlation
Summary

• Image gradients
• Seam carving – gradients as “energy”
• Gradients → edges and contours
• Template matching
  – Image patch as a filter
  – Chamfer matching
    • Distance transform

Coming up

• A1 out tonight, due in 2 weeks
• Thursday: binary image analysis
• Friday: A0 due