Computer Vision - Lecture 4

Gradients & Edges

05.11.2015

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de
leibe@vision.rwth-aachen.de
Course Outline

• Image Processing Basics
  ➢ Image Formation
  ➢ Binary Image Processing
  ➢ Linear Filters
  ➢ Edge & Structure Extraction

• Segmentation

• Local Features & Matching

• Object Recognition and Categorization

• 3D Reconstruction

• Motion and Tracking
Topics of This Lecture

• Recap: Linear Filters

• Multi-Scale representations
  - How to properly rescale an image?

• Filters as templates
  - Correlation as template matching

• Image gradients
  - Derivatives of Gaussian

• Edge detection
  - Canny edge detector
Recap: Gaussian Smoothing

- **Gaussian kernel**
  \[ G_{\sigma} = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}} \]

- **Rotationally symmetric**

- **Weights nearby pixels more than distant ones**
  - This makes sense as ‘probabilistic’ inference about the signal

- **A Gaussian gives a good model of a fuzzy blob**

Image Source: Forsyth & Ponce
Recap: Smoothing with a Gaussian

- Parameter $\sigma$ is the “scale” / “width” / “spread” of the Gaussian kernel and controls the amount of smoothing.

\[
\text{for } \sigma = 1:3:10 \\
\quad h = \text{fspecial('gaussian', fsize, sigma)}; \\
\quad \text{out} = \text{imfilter(im, h);} \\
\quad \text{imshow(out);} \\
\quad \text{pause;} \\
\text{end}
\]
Recap: Effect of Filtering

- Noise introduces high frequencies. To remove them, we want to apply a “low-pass” filter.
- The ideal filter shape in the frequency domain would be a box. But this transfers to a spatial sinc, which has infinite spatial support.
- A compact spatial box filter transfers to a frequency sinc, which creates artifacts.
- A Gaussian has compact support in both domains. This makes it a convenient choice for a low-pass filter.
Recap: Low-Pass vs. High-Pass

Original image

Low-pass filtered

High-pass filtered

Image Source: S. Chenney
Topics of This Lecture

• Recap: Linear Filters

• Multi-Scale representations
  ➢ How to properly rescale an image?

• Filters as templates
  ➢ Correlation as template matching

• Image gradients
  ➢ Derivatives of Gaussian

• Edge detection
  ➢ Canny edge detector
Motivation: Fast Search Across Scales

Image Source: Irani & Basri
Recap: Sampling and Aliasing
Recap: Sampling and Aliasing
Recap: Sampling and Aliasing

1. **Signal**
   - Fourier Transform
   - Magnitude Spectrum

2. **Sample**
   - **Sampled Signal**
     - Fourier Transform
     - Magnitude Spectrum

3. **Copy and Shift**
   - Cut out by multiplication with box filter

4. **Inaccurately Reconstructed Signal**
   - Inverse Fourier Transform
   - Magnitude Spectrum

*Image Source: Forsyth & Ponce*
Recap: Resampling with Prior Smoothing

• Note: We cannot recover the high frequencies, but we can avoid artifacts by smoothing before resampling.

Image Source: Forsyth & Ponce
The Gaussian Pyramid

\[ G_0 = \text{Image} \]

\[ G_1 = (G_0 \ast \text{gaussian}) \downarrow 2 \]

\[ G_2 = (G_1 \ast \text{gaussian}) \downarrow 2 \]

\[ G_3 = (G_2 \ast \text{gaussian}) \downarrow 2 \]

\[ G_4 = (G_3 \ast \text{gaussian}) \downarrow 2 \]

Source: Irani & Basri
Gaussian Pyramid - Stored Information

All the extra levels add very little overhead for memory or computation!
Summary: Gaussian Pyramid

- Construction: create each level from previous one
  - Smooth and sample

- Smooth with Gaussians, in part because
  - a Gaussian*Gaussian = another Gaussian
  - \[ G(\sigma_1) * G(\sigma_2) = G(\sqrt{\sigma_1^2 + \sigma_2^2}) \]

- Gaussians are low-pass filters, so the representation is redundant once smoothing has been performed.
  \[ \Rightarrow \text{There is no need to store smoothed images at the full original resolution.} \]
The Laplacian Pyramid

Gaussian Pyramid

\[ L_i = G_i - \text{expand}(G_{i+1}) \]
\[ G_i = L_i + \text{expand}(G_{i+1}) \]

Laplacian Pyramid

\[ L_n = G_n \]

Why is this useful?

Source: Irani & Basri
Laplacian ~ Difference of Gaussian

\[
\text{DoG} = \text{Difference of Gaussians}
\]

Cheap approximation - no derivatives needed.
Topics of This Lecture

- Recap: Linear Filters
- Multi-Scale representations
  - How to properly rescale an image?
- Filters as templates
  - Correlation as template matching
- Image gradients
  - Derivatives of Gaussian
- Edge detection
  - Canny edge detector
Note: Filters are Templates

- Applying a filter at some point can be seen as taking a dot-product between the image and some vector.
- Filtering the image is a set of dot products.

- Insight
  - Filters look like the effects they are intended to find.
  - Filters find effects they look like.
Where’s Waldo?

Template

Scene

Slide credit: Kristen Grauman

B. Leibe
Where’s Waldo?

Detected template

Template
Where’s Waldo?

Detected template

Correlation map

Slide credit: Kristen Grauman
Correlation as Template Matching

- Think of filters as a dot product of the filter vector with the image region
  - Now measure the angle between the vectors

\[
\mathbf{a} \cdot \mathbf{b} = \| \mathbf{a} \| \| \mathbf{b} \| \cos \theta \quad \cos \theta = \frac{\mathbf{a} \cdot \mathbf{b}}{\| \mathbf{a} \| \| \mathbf{b} \|}
\]

- Angle (similarity) between vectors can be measured by normalizing the length of each vector to 1 and taking the dot product.
Topics of This Lecture

• Recap: Linear Filters

• Multi-Scale representations
  ➢ How to properly rescale an image?

• Filters as templates
  ➢ Correlation as template matching

• Image gradients
  ➢ Derivatives of Gaussian

• Edge detection
  ➢ Canny edge detector
Derivatives and Edges...

1st derivative

Maxima of first derivative

2nd derivative

“zero crossings” of second derivative

B. Leibe
Differentiation and Convolution

- For the 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 this using finite differences:
  \[
  \frac{\partial f(x, y)}{\partial x} \approx \frac{f(x + 1, y) - f(x, y)}{1}
  \]

- To implement the above as convolution, what would be the associated filter?
  \[
  \begin{bmatrix}
  1 & -1
  \end{bmatrix}
  \]

Slide credit: Kristen Grauman
Partial Derivatives of an Image

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

Which shows changes with respect to x?

\[ \begin{bmatrix} -1 & 1 \end{bmatrix} \quad \text{or} \quad \begin{bmatrix} 1 & -1 \end{bmatrix} \]

Slide credit: Kristen Grauman
Assorted Finite Difference Filters

\[
\text{Prewitt: } \quad M_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad ; \quad M_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}
\]

\[
\text{Sobel: } \quad M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad ; \quad M_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}
\]

\[
\text{Roberts: } \quad M_x = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad ; \quad M_y = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}
\]

\[
\begin{align*}
\text{>> } & \text{My = fspecial('sobel');} \\
\text{>> } & \text{outim = imfilter(double(im), My);} \\
\text{>> } & \text{imagesc(outim);} \\
\text{>> } & \text{colormap gray;}
\end{align*}
\]

Slide credit: Kristen Grauman
Image Gradient

- The gradient of an image:
  \[ \nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \end{bmatrix} \]

- The gradient points in the direction of most rapid intensity change
  \[ \nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} & 0 \end{bmatrix} \]

- The gradient direction (orientation of edge normal) is given by:
  \[ \theta = \tan^{-1} \left( \frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial x}} \right) \]

- The edge strength is given by the gradient magnitude
  \[ \| \nabla f \| = \sqrt{\left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2} \]

Slide credit: Steve Seitz
Effect of Noise

- Consider a single row or column of the image
  - Plotting intensity as a function of position gives a signal

\[ f(x) \]

\[
\frac{d}{dx} f(x)
\]

Where is the edge?

Slide credit: Steve Seitz
Solution: Smooth First

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

Slide credit: Steve Seitz

B. Leibe
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 Filter

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

Why is this preferable?

Slide adapted from Kristen Grauman
Derivative of Gaussian Filters

- **x-direction**
- **y-direction**

Source: Svetlana Lazebnik
Laplacian of Gaussian (LoG)

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

\[
\frac{\partial^2}{\partial x^2} h
\]

\[
( \frac{\partial^2}{\partial x^2} h ) \ast f
\]

Where is the edge? Zero-crossings of bottom graph

Slide credit: Steve Seitz
Summary: 2D Edge Detection Filters

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

\[
 h_\sigma(u, v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}
\]

- Derivative of Gaussian:
  \[
  \frac{\partial}{\partial x} h_\sigma(u, v)
  \]

- Laplacian of Gaussian
Topics of This Lecture

- Recap: Linear Filters
- Multi-Scale representations
  - How to properly rescale an image?
- Filters as templates
  - Correlation as template matching
- Image gradients
  - Derivatives of Gaussian
- **Edge detection**
  - Canny edge detector
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
Designing an Edge Detector

- **Criteria for an “optimal” edge detector:**
  - **Good detection:** the optimal detector should minimize the probability of false positives (detecting spurious edges caused by noise), as well as that of false negatives (missing real edges).
  - **Good localization:** the edges detected should be as close as possible to the true edges.
  - **Single response:** the detector should return one point only for each true edge point; that is, minimize the number of local maxima around the true edge.
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
   - Thresholding, thinning

- Two issues
  - At what scale do we want to extract structures?
  - How sensitive should the edge extractor be?
Scale: Effect of $\sigma$ on Derivatives

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

$\Rightarrow$ Larger values: larger-scale edges detected
$\Rightarrow$ Smaller values: finer features detected
Sensitivity: Recall Thresholding

- Choose a threshold $t$
- Set any pixels less than $t$ to zero (off).
- Set any pixels greater than or equal to $t$ to one (on).

$$F_T[i, j] = \begin{cases} 1, & \text{if } F[i, j] \geq t \\ 0, & \text{otherwise} \end{cases}$$
Original Image

Slide credit: Kristen Grauman
Gradient Magnitude Image
Thresholding with a Lower Threshold
Thresholding with a Higher Threshold
Canny Edge Detector

- Probably the most widely used edge detector in computer vision
- Theoretical model: step-edges corrupted by additive Gaussian noise
- Canny has shown that the first derivative of the Gaussian closely approximates the operator that optimizes the product of \textit{signal-to-noise ratio} and localization.

Canny Edge Detector

1. Filter image with derivative of Gaussian
2. Find magnitude and orientation of gradient
3. Non-maximum suppression:
   - Thin multi-pixel wide “ridges” down to single pixel width
4. 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
  ```
The Canny Edge Detector

Original image (Lena)

Slide credit: Kristen Grauman
The Canny Edge Detector

Gradient magnitude

Slide credit: Kristen Grauman
The Canny Edge Detector

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

Slide credit: Kristen Grauman

B. Leibe
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 $r$
  $\Rightarrow$ Linear interpolation based on gradient direction
The Canny Edge Detector

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

Thinning
(non-maximum suppression)

Slide credit: Kristen Grauman
Solution: Hysteresis Thresholding

- Hysteresis: A lag or momentum factor
- Idea: Maintain two thresholds $k_{\text{high}}$ and $k_{\text{low}}$
  - Use $k_{\text{high}}$ to find strong edges to start edge chain
  - Use $k_{\text{low}}$ to find weak edges which continue edge chain
- Typical ratio of thresholds is roughly
  $$k_{\text{high}} / k_{\text{low}} = 2$$

Source: D. Lowe, S. Seitz
Hysteresis Thresholding

Original image

High threshold (strong edges)

Low threshold (weak edges)

Hysteresis threshold

B. Leibe

Source: L. Fei-Fei

courtesy of G. Loy
Object Boundaries vs. Edges

Background
Texture
Shadows

Slide credit: Kristen Grauman
B. Leibe
Edge Detection is Just the Beginning...

- Berkeley segmentation database:
  http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/
References and Further Reading

• Background information on linear filters and their connection with the Fourier transform can be found in Chapter 7 of F&P. Additional information on edge detection is available in Chapter 8.
  