Computer Vision - Lecture 4
Gradients & Edges
28.10.2014

Announcements
• Exercise sheet 2 is available
  • Thresholding, Morphology
  • Gaussian smoothing
  • Image gradients
  • Edge Detection
  ⇒ Deadline: Wednesday night, 05.11. (next week).
• Reminder
  • You’re encouraged to form teams of up to 3 people!
  • Make it easy for Aljosa & Dora to correct your solutions:
    • Turn in everything as a single zip archive.
    • Use the provided Matlab framework.
    • For each exercise, you need to implement the corresponding apply function. If the screen output matches the expected output, you will get the points for the exercise; else, no points.
    • Matlab helps you to find errors (red lines under your code!)

Announcements (2)
• Exam
  • There will be a written exam.
  • I’m currently organizing the exam date...
  • We’ll organize a test exam towards the end of the semester.
• Admission requirements
  • Need to reach at least 50% of the exercise points.
  • Points are given
    • for each exercise sheet.
    • for the test exam.
  • Bonus points will be available on several occasions.
  ⇒ If you follow the lecture and do the exercises regularly, you won’t have to worry about getting admitted.

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.

```matlab
for sigma=1:3:10
    h = fspecial('gaussian', fsize, sigma);
    out = imfilter(im, h);
    imshow(out);
    pause;
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

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

Recap: Sampling and Aliasing

- Sample
- Cutoff and alias
- Mag. Spectrum
- Accurately reconstructed signal
- Inexact sampling
- Inexact reconstruction
- Mag. Spectrum
- Cut out by manipulation with low filter
- Mag. Spectrum
Recap: Sampling and Aliasing

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

Recap: Resampling with Prior Smoothing

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

The Gaussian Pyramid

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.
  ⇒ There is no need to store smoothed images at the full original resolution.
The Laplacian Pyramid

\[ L_0 = G_0 - \text{expand}(G_0) \]
\[ L_i = L_{i-1} - \text{expand}(G_{i-1}) \]

Laplacian ~ Difference of Gaussian

DoG = Difference of Gaussians

Cheap approximation - no derivatives needed.

Why is this useful?

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?

Scene
Template
Detected template
Where’s Waldo?

Detected template  
Correlation map

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.
  $$a \cdot b = \frac{a}{||a||} \cdot \frac{b}{||b||} \cos \theta$$
- 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
2nd derivative
Maxima of first derivative
“Zero crossings” of second derivative

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?

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 \( x \)?

-1 1
-1 or 1 1
**Assorted Finite Difference Filters**

- **Prewitt:** 
  \[
  M_x = \begin{bmatrix}
  -1 & 0 & 1 \\
  -1 & 0 & 1 \\
  -1 & 0 & 1
  \end{bmatrix},
  M_y = \begin{bmatrix}
  -1 & -1 & -1 \\
  0 & 0 & 0 \\
  1 & 1 & 1
  \end{bmatrix}
  \]

- **Sobel:** 
  \[
  M_x = \begin{bmatrix}
  -1 & 0 & 1 \\
  -2 & 0 & 2 \\
  -1 & 0 & 1
  \end{bmatrix},
  M_y = \begin{bmatrix}
  -1 & -2 & -1 \\
  0 & 0 & 0 \\
  1 & 2 & 1
  \end{bmatrix}
  \]

- **Roberts:** 
  \[
  M_x = \begin{bmatrix}
  1 & 0 \\
  0 & -1
  \end{bmatrix},
  M_y = \begin{bmatrix}
  0 & 1 \\
  -1 & 0
  \end{bmatrix}
  \]

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

**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.

\[
\nabla f = \begin{bmatrix}
  \frac{\partial f}{\partial x} \\
  \frac{\partial f}{\partial y}
  \end{bmatrix}
\]

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

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

**Effect of Noise**

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

**Solution: Smooth First**

- Where is the edge? Look for peaks in

\[
\frac{\partial}{\partial x}(h*f)
\]

**Derivative Theorem of Convolution**

- Differentiation property of convolution.

\[
\frac{\partial}{\partial x}(h*f) = \left(\frac{\partial}{\partial x}h\right)*f
\]

**Derivative of Gaussian Filter**

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

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

Why is this preferable?
Derivative of Gaussian Filters

- $x$-direction
- $y$-direction

Laplacian of Gaussian (LoG)

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

Where is the edge? Zero-crossings of bottom graph

Summary: 2D Edge Detection Filters

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

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 $\rightarrow$ 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).

Thresholding with a Lower Threshold

Original Image

Gradient Magnitude Image
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 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:

  ```matlab
  >> edge(image, 'canny');
  >> help edge
  ```

The Canny Edge Detector

Original image (Lena)
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
  - Linear interpolation based on gradient direction

The Canny Edge Detector

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

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 $k_{\text{high}} / k_{\text{low}} = 2$

Object Boundaries vs. Edges

- 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/)
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.