Digital Image Processing: Sharpening Filtering in Frequency Domain

CSC621-821
Biomedical Imaging and Analysis
Dr. Kazunori Okada

Overview

• Last lecture
  – Practical Foundation of Digital Image Processing I & II
    – Spatial Domain Analysis (Image Filtering)
    – Point Processing: Intensity Transformations
    – Neighbor Processing: Spatial Smoothing Filtering
    – Smoothing
    – Derivative Filtering
    – Sharpening Filtering in Spatial Domain (Laplace)

• Today’s lecture
  – Practical Foundation of Digital Image Processing III
    – Edge Detection in Spatial Domain
    – Multiple-Image Operation
    – Frequency Domain Processing
Edge Detection

- Another fundamental IP task
- Edge detection finds discontinuity of gray levels in images
- For object recognition
  - Outline of object shape
  - Texture patterns
- For segmentation
  - Boundary between regions
- Edge detection is also based on spatial filtering using derivatives.

What is Edges?

- An edge is a set of connected pixels that lie on the boundary between two regions
**Edge Detection by Derivative Filters**

- We have already spoken about how derivatives are used to find discontinuities.
- 1\textsuperscript{st} derivative tells us where an edge is.
- 2\textsuperscript{nd} derivative can be used to show edge direction.

**1\textsuperscript{st} vs 2\textsuperscript{nd} Derivative Comparison**

- Comparing the 1\textsuperscript{st} and 2\textsuperscript{nd} derivatives we can conclude the following:
  - 1\textsuperscript{st} order derivatives generally produce thicker edges.
  - 2\textsuperscript{nd} order derivatives have a stronger response to fine detail e.g. thin lines.
  - 1\textsuperscript{st} order derivatives have stronger response to gray level step.
  - 2\textsuperscript{nd} order derivatives produce a double response at step changes in grey level.
1st Derivatives for Filtering

- Implementing 1st derivative filters is difficult in practice.
- For a function $f(x, y)$, the gradient of $f$ at coordinates $(x, y)$ is given as the column vector:

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

1st Derivative Filtering Cond.

- The magnitude of this vector is given by:

$$\| \nabla f \| = \text{mag}(\nabla f) = \left[ G_x^2 + G_y^2 \right]^{1/2} = \left[ \left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2 \right]^{1/2}$$

- For practical reasons this can be simplified as:

$$\| \nabla f \| \approx |G_x| + |G_y|$$
1st Derivative Filtering Cond.

- There is some debate as to how best to calculate these gradients but we will use:
  \[ \| \nabla f \| \approx |(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)| \]
  \[ + |(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)| \]
- which is based on these coordinates

Digital Filter: Sobel Filter

- Based on the previous equations we can derive the **Sobel Operators**

\[
\begin{array}{ccc}
-1 & 0 & 1 \\
-2 & 0 & 2 \\
-1 & 0 & 1 \\
\end{array}
\]

- To filter an image it is filtered using both operators the results of which are added together
Sobel Filter Example

An image of a contact lens which is enhanced in order to make defects (at four and five o’clock in the image) more obvious.

Edge Detector Example

Original Image

Vertical Gradient Component

Horizontal Gradient Component

Combined Edge Image
Edge Detector: Original

Edge Detector: Horizontal
Edge Detector: Vertical

Edge Detector: Combined
Issues for Edge Detection

• Often, problems arise in edge detection in that there is **too much detail**
• For example, the brickwork in the previous example
• One way to overcome this is to **smooth images prior to edge detection**

Sobel Filter with Smoothing

<table>
<thead>
<tr>
<th>Original Image</th>
<th>Vertical Gradient Component</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Original Image" /></td>
<td><img src="image2.png" alt="Vertical Gradient Component" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Horizontal Gradient Component</th>
<th>Combined Edge Image</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3.png" alt="Horizontal Gradient Component" /></td>
<td><img src="image4.png" alt="Combined Edge Image" /></td>
</tr>
</tbody>
</table>
Other Common Edge Detectors

• Given a 3*3 region of an image the following edge detection filters can be used

<table>
<thead>
<tr>
<th></th>
<th>z_1</th>
<th>z_2</th>
<th>z_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>z_4</td>
<td>z_5</td>
<td>z_6</td>
<td></td>
</tr>
<tr>
<td>z_7</td>
<td>z_8</td>
<td>z_9</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>-1</th>
<th>-1</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Prewitt

<table>
<thead>
<tr>
<th></th>
<th>-1</th>
<th>-2</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Roberts

<table>
<thead>
<tr>
<th></th>
<th>-1</th>
<th>0</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>-1</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Subel

Issues for Derivative Filtering

• Derivative based edge detectors are extremely sensitive to noise
Issues for Derivative Filtering

- The higher-order derivative becomes more susceptible to image noises

\[ I(x) \]

- Solution: Smoothed Derivative Filters
  - Sharpening by derivative filtering
  - Smoothing by Gaussian filtering

Gaussian Smoothing Filter

- A very common smoothing filter
- A weighted averaging in a discrete form
- Used because
  - Smooth (infinitely differentiable)
  - Decay to zero rapidly
  - Simple analytic formula
  - Separable: multidimensional Gaussian = product of Gaussians in each dimension
  - Convolution of 2 Gaussians = Gaussian
  - Limit of applying multiple filters is Gaussian (Central limit theorem)

\[ g(x) \]
Gaussian Filters

- Continuous ($\sigma=1$)

$$G(x) = \frac{1}{\sqrt{2\pi \sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$

- Discrete ($\sigma=1$)

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

How to derive the filter?

1. Distance size 3x3? 5x5? $3 \times 3$?
2. Distance $\sigma$ (parameter) $\sigma = 2$
3. Distance (coordinate space)
   - Define coordinate of each cell. (-1,1)
4. For each cell, compute $G(x, y)$:
   - $(1,1) \rightarrow G(1,1) = \frac{1}{2\pi \sigma^2} e^{-\frac{1}{2\sigma^2}}$
   - $= \frac{1}{2\pi \cdot 2^2} \cdot \frac{1}{8\pi}$
5. Do this for all cells
6. Round #s & normalize them.

Canny Edge Detection

- **Optimal** edge detector
  - Detection
  - Localization
  - Response reduction

- **Four step process**
  - Gaussian smoothing (previous slide)
  - 1st Derivative edge detection (e.g., sobel)
    - Magnitude
    - Orientation
  - Non maximum suppression
  - Edge tracing by thresholding with hysteresis
    - Two thresholds

\[ G = \sqrt{G_x^2 + G_y^2} \]
\[ \Theta = \arctan\left(\frac{G_y}{G_x}\right) \]

Canny Operator Example: Break

Original Image  Edges
2nd Derivative: Recall Laplace?

- We encountered the 2nd-order derivative based Laplace filter already.

\[
\begin{array}{ccc}
0 & -1 & 0 \\
-1 & 4 & -1 \\
0 & -1 & 0 \\
\end{array}
\quad
\begin{array}{ccc}
-1 & -1 & -1 \\
-1 & 8 & -1 \\
-1 & -1 & -1 \\
\end{array}
\]

- The Laplacian is typically not used by itself as it is too sensitive to noise.
- Usually, for edge detection, the Laplacian is combined with a smoothing Gaussian filter.

Smoothed Derivative

- Smoothing convolution filtering (*) and derivative filtering \((d/dx)\) commute: \(f\): image, \(g\): smoothing filter.

\[
\frac{d}{dx}(f * g) = \frac{df}{dx} * g = f * \frac{dg}{dx}
\]

- Smooth-then-gradient = gradient-then-smooth.
- **Combined filter**: \(dg/dx\) is a smoothed derivative.
  - 1st derivative of Gaussian can be analytically computed.
Laplacian of Gaussian (LoG)

- The Laplacian of Gaussian (or LoG, Mexican hat): 2nd derivative of 2D Gaussian smoothing filter for detecting edges with noise removal.

\[
\text{LoG}(x,y) = -\frac{1}{\pi\sigma^4} \left[ 1 - \frac{x^2 + y^2}{2\sigma^2} \right] e^{-\frac{x^2+y^2}{2\sigma^2}}
\]

Zero Crossing

Find peaks/maximum of gradient image

Find pixels with zero value & change of sign

Zero-Crossing

http://www.pages.drexel.edu/~weg22/edge.html
**Multiple Image Operation: Overview**

- **Multiple Image Operation**
  1. Given a $M$ number of images with the same dimension/size and types ($M \geq 2$)
  2. Produce an image by manipulating the $M$ images

- **Important tool for**
  - Performing combination of multiple filters (Laplace!)
  - Detecting region of interest (Angiology)
  - Reducing noise
  - Studying image statistics!
**Image Addition**

- Adding two or more images pixel-wise
- Rescale intensity range if out of bounds

\[ I_+ (x, y) = I_1 (x, y) + I_2 (x, y) \forall x, y \]

**Image Subtraction**

- Subtracting two or more images pixel-wise
- Rescale intensity range if out of bounds

\[ I_- (x, y) = I_1 (x, y) - I_2 (x, y) \forall x, y \]
**Image Multiplication**

- Multiplying two or more images pixel-wise
- Rescale intensity range if out of bounds
- Masking
- Will be used in Fourier Transformation (Next Part)

\[ I_\ast(x, y) = I_1(x, y) \ast I_2(x, y) \forall x, y \]

**Average Image**

\[ I_{av}(x, y) = \frac{1}{M} (I_1(x, y) + \ldots + I_M(x, y)) \]

Important for statistical analyses for a set of images…
For this they must be aligned well
Registration!!!
Combination of Spatial Filtering

- Successful image enhancement is typically not achieved using a single operation!!!
- Rather we combine a range of techniques in order to achieve a final result
- This example will focus on enhancing the bone scan to the right

**Combination 1**

(a) Laplacian filter of bone scan (a)
(b) Sharpened version of bone scan achieved by subtracting (a) and (b)
(c) Sobel filter of bone scan (a)
(d)
Combination 2

- Image (d) smoothed with a 5x5 averaging filter
- The product of (c) and (e) which will be used as a mask
- Sharpened image which is sum of (a) and (f)
- Result of applying a power-law trans. to (g)

Final Result

- Compare the original and final images
Frequency Domain Analysis

Two equivalent ways to get the same result!!!

Spatial Domain

Convolution filtering

Fourier Transformation

Inverse Fourier Transformation

Frequency Domain

Product filtering

Fourier Series

Any function that periodically repeats itself can be expressed as a sum of sines and cosines of different frequencies each multiplied by a different coefficient – a Fourier series
Notice how we get closer and closer to the original function as we add more and more frequencies.

Fourier Series Formula

• Periodic function \( f(x) \) defined over \([-\pi .. \pi]\)

\[
f(x) = \frac{1}{2} a_0 + \sum_{n=1}^{\infty} a_n \cos(nx) + b_n \sin(nx)
\]

where

\[
a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos(nx) \, dx
\]

\[
b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin(nx) \, dx
\]
Fourier Transformation: Idea

- Transform applied to function to analyze its “frequency” content
- Several versions
  - Fourier series:
    - input = continuous, bounded; output = discrete, unbounded
  - Fourier transform:
    - input = continuous, unbounded; output = continuous, unbounded
  - Discrete Fourier transform (DFT):
    - input = discrete, bounded; output = discrete, bounded
- Fourier Transform is reversible (inverse)

Fast Fourier Transformation

- The reason that Fourier based techniques have become so popular is the development of the Fast Fourier Transform (FFT) algorithm
- Allows the Fourier transform to be carried out in a reasonable amount of time
- Reduces the amount of time required to perform a Fourier transform by a factor of 100 – 600 times!
Discrete Fourier Transform (DFT)

- The DFT of a two dimensional image can be visualised by showing the spectrum of the images component frequencies.

2D Frequency Space
Example: Sine Curves

Example: Spatial Domain
Example: Frequency Domain

Example: Complex Scenes
DFT Examples

Scanning electron microscope image of an integrated circuit magnified ~2500 times

Fourier spectrum of the image

Original Image
DFT-based Image Processing: Overview

- To filter an image in the frequency domain:
  1. Compute $F(u, v)$ the DFT of the image
  2. Multiply $F(u, v)$ by a filter function $H(u, v)$
  3. Compute the inverse DFT of the result
**What happen in Frequency Domain?**

- Fourier transform turns convolution into multiplication:
  \[ F(f(x) * g(x)) = F(f(x))F(g(x)) \]

- Image filtering in Frequency Domain is a MULTIPLICATION! (Faster!)

**Convolution Theorem**

- \[ F(f(x) * g(x)) = F(f(x))F(g(x)) \]

- Efficient computation
  - Spatial Filtering by convolution is time consuming \( O(n^2) \)
  - Fast Fourier Transform (FFT) takes time \( O(n \log n) \)
  - Thus, convolution can be performed in time \( O(n \log n + m \log m) \)
  - Greatest efficiency gains for large filters
Basic Frequency Filters

Low Pass Filter

High Pass Filter

Low Pass Filtering
High Pass Filtering

Smoothing by Low Pass Filtering

- Smoothing is achieved by dropping out the high frequency components
- The basic model for filtering is:
  - $G(u,v) = H(u,v)F(u,v)$
  - $F(u,v)$ is the Fourier transform of the image being filtered
  - $H(u,v)$ is the transfer function
- Low pass filters – only pass the low frequencies, drop the high ones
Ideal Filter

- Simply cut off all high frequency components that are a specified distance $D_0$ from the origin of the transform.

- Varying the distance $D_0$ changes the behaviour of the filter.

Transfer Function

- The transfer function for the ideal low pass filter can be given as:

$$H(u, v) = \begin{cases} 
1 & \text{if } D(u, v) \leq D_0 \\
0 & \text{if } D(u, v) > D_0 
\end{cases}$$

- where $D(u, v)$ is given as:

$$D(u, v) = \left[\left(u - \frac{M}{2}\right)^2 + \left(v - \frac{N}{2}\right)^2\right]^{1/2}$$
Ideal Filter Example

- Above we show an image, its Fourier spectrum and a series of ideal low pass filters of radius 5, 15, 30, 80 and 230 superimposed on top of it.

Five Radii
Gaussian Filter

- The transfer function of a Gaussian low-pass filter is defined as:

\[ H(u, v) = e^{-D^2(u,v)/2D_0^2} \]
Gaussian Filter Results

Original image

Result of filtering with Gaussian filter with cutoff radius 5

Result of filtering with Gaussian filter with cutoff radius 30

Result of filtering with Gaussian filter with cutoff radius 230

Result of filtering with Gaussian filter with cutoff radius 85

Low Pass Filter Comparison

Result of filtering with ideal low pass filter of radius 15

Result of filtering with Gaussian filter with cutoff radius 15
Example: Texts

Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

Example: Faces
Sharpening by High Pass Filtering

• Edges and fine detail in images are associated with high frequency components
• *High pass filters* – only pass the high frequencies, drop the low ones
• High pass frequencies are precisely the reverse of low pass filters, so:
  \[ H_{hp}(u, v) = 1 - H_{lp}(u, v) \]

Ideal High-Pass Filter

• The ideal high pass filter is given as:
  \[ H(u,v) = \begin{cases} 
  0 & \text{if } D(u,v) \leq D_0 \\
  1 & \text{if } D(u,v) > D_0 
\end{cases} \]
• where \( D_0 \) is the cut off distance as before
Ideal Filter Results

Results of ideal high pass filtering with $D_0 = 15$

Results of ideal high pass filtering with $D_0 = 30$

Results of ideal high pass filtering with $D_0 = 80$

Gaussian Filter

- The Gaussian high pass filter is given as:

$$H(u, v) = 1 - e^{-D^2(u, v)/2D_0^2}$$

- where $D_0$ is the cut off distance as before
Gaussian Filter Results

Results of Gaussian high pass filtering with $D_0 = 15$

Results of Gaussian high pass filtering with $D_0 = 80$

Results of Gaussian high pass filtering with $D_0 = 30$

High Pass Filter Comparison

Results of ideal high pass filtering with $D_0 = 15$

Results of Gaussian high pass filtering with $D_0 = 15$
Sampling for Discritization

• Can’t store continuous signal: instead store “samples”
  – Usually evenly sampled:
    \[ f_0 = f(x_0), f_1 = f(x_0 + \Delta x), f_2 = f(x_0 + 2\Delta x), f_3 = f(x_0 + 3\Delta x), \ldots \]

• Instantaneous measurements of continuous signal
  – This can lead to problems

Nyquist-Shannon Sampling Theorem

• If a function contains no frequency higher than \( B \) Hz, it can be perfectly reconstructed from infinite sequence of samples when the sample rate is higher than \( 2B \)

• Given a discrete image at sampling rate \( B \), image details whose frequency is higher than \( \frac{1}{2} B \) (Nyquist/cut-off frequency) cannot be retained.
Down Sampling / Gaussian Pyramid

• By factor of 2
• Take every two pixels, volume becomes 1/4

Gaussian Pyramid
- for reducing size
- for robust representation

Aliasing Problem

• Problem: Image gets increasingly noisy!!!
• Reason is called **Aliasing**:
  – When down-sample an image, all image frequencies get doubled
  – This can push some high frequency details above the cut-off frequency!
  – Those lost information become NOISE!
• Solution: perform low-pass filter every time down-sample image
  – Get rid of higher frequency that goes beyond cut-off
  – **Blur the image first then down sample!!!**
Summary

- Practical Foundation of Digital Image Processing III
- Sharpening Filtering in Spatial Domain
- Edge Detection in Spatial Domain
- Filter Combination
- Multiple-Image Operation
- Frequency Domain Techniques

- Next Week:
  - Advanced Image Processing
  - Edge-Preserving Smoothing
  - Morphological Operations
  - Connected Component Analysis