ELIC 629
Digital Image Processing

Winter 2006

Convolution Theorem, Discontinuity Detection and Image Segmentation

Bill Kapralos

Tuesday, April 18 2006

Overview (1):

- **Before We Begin**
  - Administrative details
  - Review → some questions to consider
- **Convolution Theorem**
  - Brief review from last week
  - Properties
  - Gaussian filters

Overview (2):

- **Discontinuity Detection**
  - Introduction to image segmentation
  - Point detection
  - Line detection
  - Edge detection
- **Thresholding**
  - Foundation
  - Introduction

Before We Begin

Administrative Details (1):

- **Lab Eight Today**
  - Final lab
  - No lab report required
  - Lab does include an assignment
    - Assignment is very good practice for your exam – I recommend you complete it!

Administrative Details (2):

- **Final Exam April 25 2006**
  - Short review at the end of this lecture
    - I will make some comments regarding the exam
  - Exam will be similar in format to mid-term
    - No surprises!
  - Focus on material after mid-term but you are still responsible for all material
    - Still need to know filtering in the spatial domain

Some Questions to Consider (1):

- Why filter in the frequency domain ?
- What are the steps to filtering an image in the frequency domain ?
- Why do we shift the origin of the DFT output ?
- From the origin, what can we say about the DFT frequency ?
- What is a low/high pass frequency domain filter ?
- What is a "notch" filter ?

Convolution Theorem

Frequency vs. Spatial Domain (1):

- **Convolution Theorem**
  - Establishes most fundamental relationship between frequency and spatial domains
    - Remember filtering in the spatial domain ?
    - Formally, convolution of two functions denoted by f(x,y) * h(x,y) is defined by

$$f(x,y)*h(x,y)=\frac{1}{MN}\sum_{0}^{M-1}\sum_{0}^{N-1}f(m,n)h(x-m,y-n)$$

  - Minus sign in h(x-m, y-n) means that the function h is mirrored about the origin
    - Inherent in the definition of convolution

Frequency vs. Spatial Domain (2):

- **Convolution Theorem (cont…)**

$$f(x,y)*h(x,y)=\frac{1}{MN}\sum_{0}^{M-1}\sum_{0}^{N-1}f(m,n)h(x-m,y-n)$$

  - Basically, above equation states the following
    1. Flipping one function about the origin
    2. Shifting that function with respect to the other by changing the values of (x, y)
    3. Computing a sum of products over all values of m and n for each displacement (x, y) → displacements (x, y) are integer increments that stop when the functions no longer overlap

Frequency vs. Spatial Domain (3):

- **Convolution Theorem (cont…)**
  - Consider the following definitions
    - F[u, v] → Fourier transform of f[x,y]
    - H[u, v] → Fourier transform of h[x,y]
  - One half of the convolution theorem states that f(x,y)*h(x,y) and F[u,v]H[u,v] comprise a Fourier transform pair. Mathematically,

$$f(x,y)*h(x,y) \Leftrightarrow F[u,v]H[u,v]$$

  - In words, convolution in the spatial domain is equal to multiplication in the frequency domain

Frequency vs. Spatial Domain (4):

- **Convolution Theorem (cont…)**
  - Other half of the convolution theorem states that f(x,y)h(x,y) and F[u,v]*H[u,v] comprise a Fourier transform pair. Mathematically,

$$f(x,y)h(x,y) \Leftrightarrow F[u,v]*H[u,v]$$

  - In words, multiplication in the spatial domain is equal to convolution in the frequency domain

Frequency vs. Spatial Domain (5):

- **Filters in the Spatial and Frequency Domain Form a Fourier Transform Pair**
  - Given filter in frequency domain to obtain filter in spatial domain
    - Take inverse DFT of the frequency domain representation of the filter
  - Given filter in spatial domain to obtain filter in frequency domain
    - Take DFT of the spatial domain representation of the filter

Frequency vs. Spatial Domain (6):

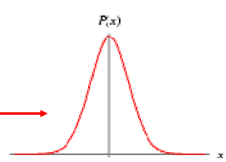
- **Some Notes**
  - All functions previously described are of size M x N (e.g., images and frequency domain representation)
    - Given same size filters in both spatial and frequency domains, typically more computationally efficient to filter in frequency domain
    - But not always worth taking DFT of spatial domain function to get frequency domain rep.

Gaussian Filters (1):

- **What is a Gaussian Function ?**
  - Normal distribution with mean μ and variance σ^2
  - Defined by the following distribution function

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

Graphical illustration of the Gaussian distribution - 1D



Gaussian Filters (2):

- **Gaussian Function as a Filter**
  - Very Useful and important
    - Their shape is easily specified
    - Both DFT and IDFT of a Gaussian is also a Gaussian
    - An averaging ("blurring") filter

Gaussian Filters (3):

- **Mathematically → Fourier Transform Pair**
  - Let H[u] denote frequency domain Gaussian filter given by

$$H[u] = Ae^{-u^2/2\sigma^2}$$

  - Corresponding filter in the spatial domain is given by

$$h[x] = \sqrt{2\pi}\sigma A e^{-2\pi^2\sigma^2 x^2}$$

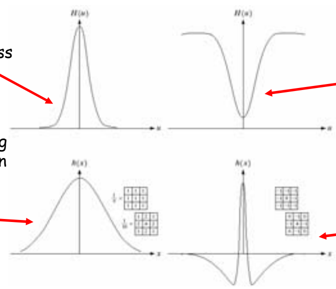
  - Both functions above comprise a Fourier transform pair → both Gaussian and real valued (e.g., no complex numbers!

Gaussian Filters (4):

- **Fourier Transform Pair (cont…)**
  - Both functions behave reciprocally to each other
    - When H[u] has a broad profile (and therefore large σ) → h[x] will have a narrow profile
    - When h[x] has a broad profile (and therefore large σ) → H[u] will have a narrow profile

Gaussian Filters (5):

- **Graphical Examples of Gaussian Filters**



Discontinuity Detection

Image Segmentation (1):

- **What is Image Segmentation ?**
  - Segmentation sub-divides an image into a number of regions or objects
  - How far this sub-division is carried out depends on the task
  - An extremely difficult yet important task
    - Its accuracy determines the eventual success or failure of any automated analysis procedure which rely on its output

Image Segmentation (2):

- **Image Segmentation Algorithms Generally Based on Two Basic Properties of Intensity**
  - Discontinuity
    - Partition image based on abrupt changes in intensity (e.g., edges where there is a large change in intensity between adjacent pixels)
  - Similarity
    - Partition image into regions that are similar based on some pre-defined criteria (e.g., intensity of pixels within a certain range)

Introduction (1):

- **Will Focus on Three Types of Discontinuities**
  1. Points
  2. Lines
  3. Edges
  - Regardless the type of discontinuity, most common approach to locating them is to “filter” the image with a 3 x 3 mask (e.g.,, convolution)
    - Mask coefficients are chosen depending on the type of discontinuity being searched for

Introduction (2):

- **Recall Spatial Domain Filtering with Mask**
  - Sum of products of coefficients with the gray levels in image encompassed by the mask

R = w(-1,-1)f(x-1,y-1) + w(-1,0)f(x-1,y) + … + w(0,0)f(x,y) +…+ w(1,0)f(x+1,y) + w(1,1)f(x+1,y+1)

w(-1,-1)	w(-1,0)	w(-1,1)
w(0,-1)	w(0,0)	w(0,1)
w(1,-1)	w(1,0)	w(1,1)

Example of a 3x3 template with its coefficients

Point Detection (1):

- **In Principle, Straightforward**
  - Using the following mask, a point is detected at the location at which the mask is centered on if

$$|R| \geq T$$

R → output of filtering operation (e.g., sum of filter coefficients multiplied by corresponding image intensities)

T → threshold (an intensity value, recall your labs)

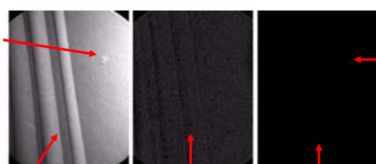
−1	−1	−1
−1	8	−1
−1	−1	−1

Point Detection (2):

- **Basic Idea**
  - Isolated point (a point whose gray level is much different from its background) will be different from its surroundings and will be detected by the mask used
  - Examine mask coefficients
    - Sum of coefficients equals 0 → mask response will be zero in areas of constant gray level

Point Detection (1):

- **Graphical Example**



Line Detection (1):

- **More Difficult Than Point Detection**
  - Lines can be oriented in any manner (e.g., horizontally, vertically, +/-45^{0}, etc.)
    - Different mask to detect each line orientation

−1	−1	−1
2	2	2
−1	−1	−1

Horizontal

−1	−1	2
−1	2	−1
2	−1	−1

+45°

−1	2	−1
−1	2	−1
−1	2	−1

Vertical

2	−1	−1
−1	2	−1
−1	−1	2

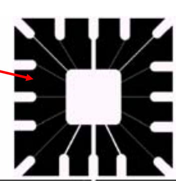
−45°

Line Detection (2):

- **Notes Regarding the Line Detection Masks**
  - Typically these masks detect lines 1 pixel thick
  - Preferred direction of each mask is weighted with a larger coefficient than the other possible directions (e.g., 2 instead of -1)
  - Coefficients sum to zero
    - Response will be equal to zero in areas of constant gray level

Line Detection (3):

- **Line Detection Graphical Example**

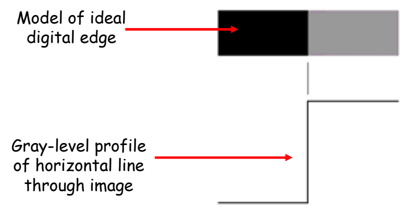


Edge Detection (1):

- **Basic Formulation**
  - What is an edge (review) → set of connected pixels that lie on a boundary between two regions
    - Different from a boundary → boundary is more of a "global" concept whereas edge is a more of a "local" concept
  - Modeling of an ideal edge
    - A set of connected pixels, each of which is located at an orthogonal step transition in gray level

Edge Detection (2):

- **Basic Formulation (cont…)**
  - Modeling of ideal edge – graphical illustration

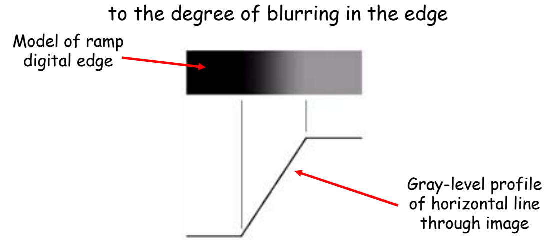


Edge Detection (3):

- **In Practice, Ideal Edges Do Not Exist!**
  - Sampling and the fact that sampling acquisition equipment etc. is far from perfect leads to edges that are blurred
  - Changing illumination (lighting conditions) will cause changes to edges & all parts of an image in general
    - Changes in lighting is actually a HUGE problem for vision/image processing tasks → many algorithms will not generalize across different lighting conditions
    - Color constancy → a big field in computer vision but still an un-solved problem!

Edge Detection (4):

- **In Practice, Ideal Edges Dont Exist! (cont..)**
  - In reality, edges have a more "ramp-like" profile
    - The slope of the ramp is inversely proportional to the degree of blurring in the edge

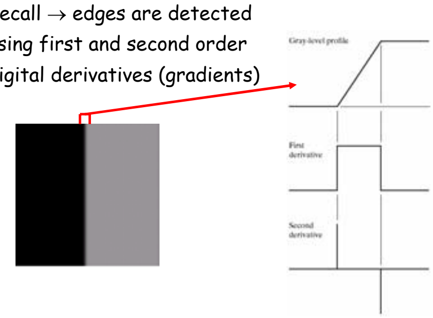


Edge Detection (5):

- **In Practice, Ideal Edges Dont Exist! (cont..)**
  - Edge is no longer a one-pixel thick path
    - An edge point is now any point contained in the ramp and an edge would be a set of such points which are connected
    - Thickness of edge is given by length of ramp which is determined by the slope which itself is determined by the amount of blurring
    - Blurred edges are typically thicker e.g., the greater the blurring → the thicker the edge

Edge Detection (6):

- **Detecting Edges**
  - Recall → edges are detected using first and second order digital derivatives (gradients)



Edge Detection (7):

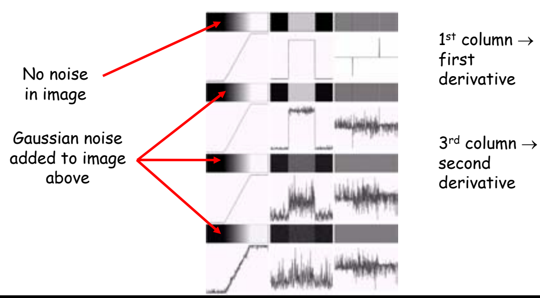
- **Detecting Edges (cont…)**
  - Remember
    - First derivative → positive at points of transition into and out of ramp (moving from left to right) & zero in constant gray-level areas
    - Second derivative → positive at transition associated with the "dark" side of edge, negative at light side of edge and zero along ramp & in areas of constant gray level

Edge Detection (8):

- **Detecting Edges (cont…)**
  - Some conclusions regarding derivatives & edges
    - Magnitude of first order derivative can be used to detect presence of edge at point
    - Sign of second order derivative can be used to determine whether edge pixel itself lies on dark or bright side of edge
    - Second order derivative produces two values for every edge & therefore zero-crossing
    - Zero-crossing → imaginary straight line drawn from positive to negative value would cross zero near midpoint of the edge

Edge Detection (9):

- **Edge Detection Example**
  - Entire transition from left to right is single edge



Edge Detection (10):

- **Edge Detection Example**
  - Conclusions we can draw from previous examples
    - To be classified as edge point, gray-level transition must be significantly stronger than background
    - Threshold used to determine whether it is different from background → e.g., will be classified as edge only if derivative is greater than some but thresholds have their own problems!
    - The set of all these points greater than the threshold and connected comprise the edge

Thresholding

Introduction (1):

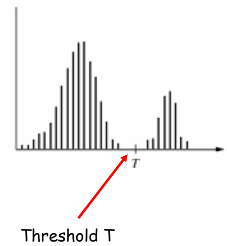
- ***Central to Image Processing/Computer Vision***
  - Essentially, thresholding basically involves performing a check at each pixel location
  - This should be familiar from your labs!

For each pixel (x,y) in image

1. Obtain pixel intensity p_i
2. Compare p_i with pre-defined threshold value T
   - if $p_i \geq T$ then $p_i = 1$ (p_i is an object point)
   - if $p_i < T$ then $p_i = 0$ (p_i is background point)

Introduction (2):

- **Graphical Example**
  - Histogram of image with light object and dark background
  - After performing thresholding of image with threshold T, pixels corresponding to object will be highlighted (e.g., set to 1) while background pixels will be set to zero



Introduction (3):

- **Multi-Level Thresholding**
  - Can be used to locate (detect) multiple objects where each object is within some range of intensities
    - Multiple thresholds and therefore multiple checks per pixel
    - For example, two objects, two threshold T_1, T_2
    - Pixel belongs to one object if $T_1 < f(x,y) \leq T_2$
    - Pixel belongs to other object if $f(x,y) > T_2$
    - Pixel belongs to background if $f(x,y) \leq T_1$

Introduction (4):

- **Graphical Example**
  - Multi-level thresholding