7-9 Monday – 309-GD2

Xử lý ảnh
INT3404 1

Giảng viên: TS. Nguyễn Thị Ngọc Diệp

Email: ngocdiep@vnu.edu.vn

Slide & code: https://github.com/chupibk/INT3404_1

Lịch trình

Tuần	Nội dung	Yêu cầu đối với sinh viên
1	Giới thiệu môn học Làm quen với OpenCV + Python	Cài đặt môi trường: Python 3, OpenCV 3, Numpy, Jupyter Notebook
2	Phép toán điểm (Point operations) – Điều chỉnh độ tương phản – Ghép ảnh	Làm bài tập 1: điều chỉnh gamma tìm contrast hợp lý
3	Histogram - Histogram equalization - Phân loại ảnh dùng so sánh histogram	Thực hành ở nhà
4	Phép lọc trong không gian điểm ảnh (linear processing filtering) - làm mịn, làm sắc ảnh	Thực hành ở nhà Tìm hiểu thêm các phép lọc
5	Tìm cạnh (edge detection)	Thực hành ở nhà
6	Các phép toán hình thái (Erosion, Dilation, Opening, Closing) - tìm biển số	Làm bài tập 2: tìm barcode
7	Chuyển đổi không gian - miền tần số (Fourier) - Hough transform	Thực hành ở nhà
8	Phân vùng (segmentation) - depth estimation - threshold-based - watershed/grabcut	Đăng ký thực hiện bài tập lớn
9	Mô hình màu Chuyển đổi giữa các mô hình màu	Làm bài tập 3: Chuyển đổi mô hình màu và thực hiện phân vùng
10	Mô hình nhiễu -Giảm nhiễu -Khôi phục ảnh -Giảm nhiễu chu kỳ - Ước lượng hàm Degration -Hàm lọc ngược, hàm lọc Wiener	Thực hành ở nhà
11	Template matching – Image Matching	Làm bài tập 4: puzzle
12	Nén ảnh	Thực hành ở nhà
13	Hướng dẫn thực hiện đồ án môn học	Trình bày đồ án môn học
14	Hướng dẫn thực hiện đồ án môn học	Trình bày đồ án môn học
15	Tổng kết cuối kỳ	Ôn tập

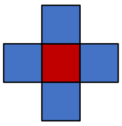
Ôn lại tuần 4: Phép toán trên điểm ảnh

- Biến đổi trên điểm ảnh (e.g., gamma)
- Ghép ảnh (cộng, trừ, tính trung bình)
- Tính toán và sử dụng histogram
- Bài tập về nhà:
  - Deadline: 15/9/2019 23:59 → 45 – 5 (late submission) = 40
  - https://forms.gle/ALrnAWc3c6h3dMqU9
  - Bài "vớt vát":
    - https://forms.gle/bNTMBqKuuBg4V4o3A

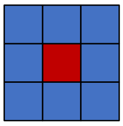
Recall week 4 & some additions

Mask shape (1/2)

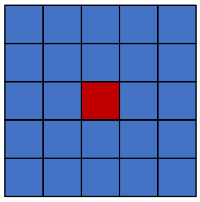
- Giá trị của điểm ảnh đầu ra tại một vị trí cụ thể phụ thuộc vào giá trị của các điểm ảnh xung quanh



4-neighbor

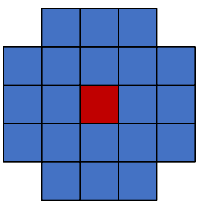


8-neighbor
Mask size 3x3

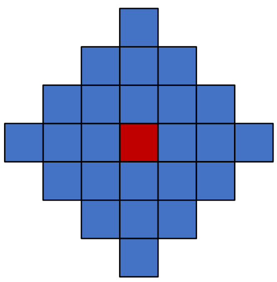


Mask size 5x5

Mask shape (2/2)



Round with radius of 3



Lozenge shape

Các phép lọc thường gặp

- Smoothing and noise removal
  - Average filtering
  - Median filtering
  - Gaussian filtering
- Edge detection and highboost filtering
  - Gradient methods
  - Laplace method
  - Highboost filtering

Week 4!

Ôn lại: Average filter

- To replace the value of every pixel by the average of the gray levels in the neighborhood defined by the filter mask
- The larger the size of the filter, the more significant blurring observed

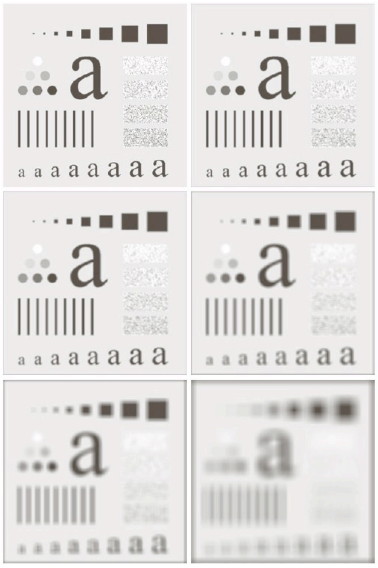
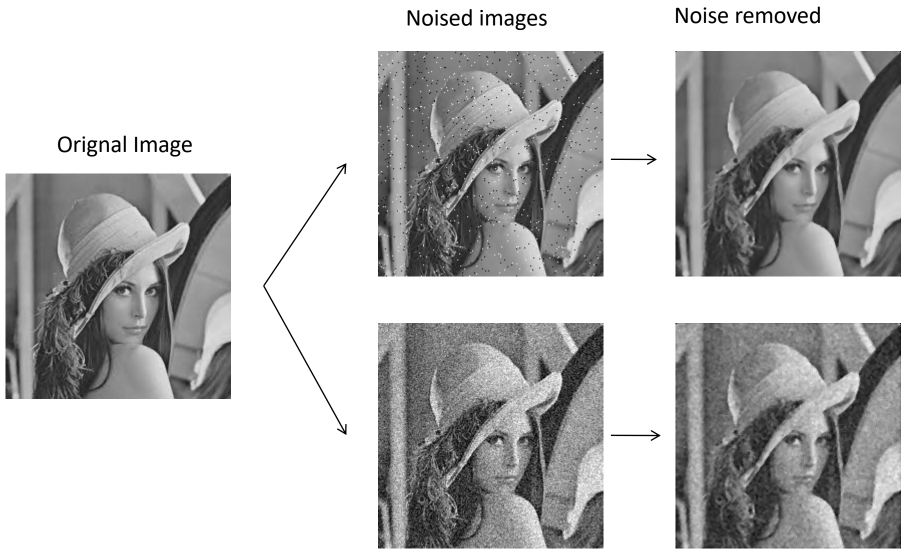


Image credit: Gonzalez et. al.

Ôn lại: Median filter != Average filter

- A = {5 4 1 6 2 7 2 6 2 10 2 3}
- The median value:
  - Sort array A = [1 2 2 2 2 3 4 5 6 6 7 10]
  - Choose the center value A = [1 2 2 2 2 3 4 5 6 6 7 10]
  - The median value is 3
- The average value is 4.667

Median filter: best for salt & pepper noise



Gradient

Edges

- An edge = a significant local change in the image intensity

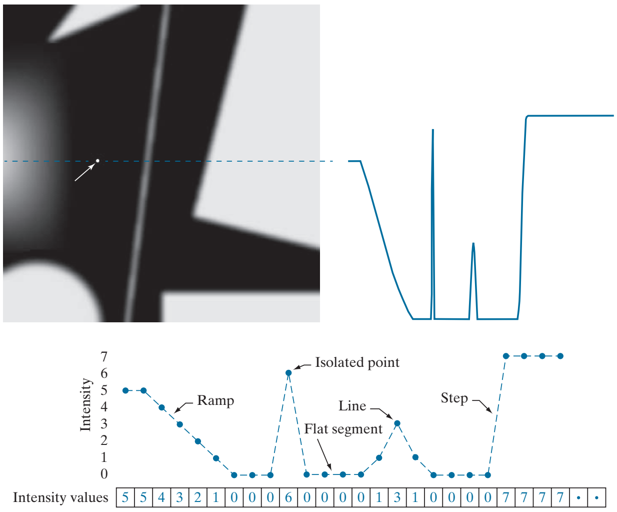


Image credit: Gonzalez et.al., Fig. 10.2

Gradient

- Gradient of a function indicates how strong the function increases.
  - For 1-dimension function: $f(x) = x^2$

$$Grad(x) = \frac{\partial f(x)}{\partial(x)} = 2x$$

  - Grad(2)=4 indicates the the increasing direction of the function is to the right.
  - Grad(-1)=-2 indicates the increasing direction of the function is to the left.

First-order and second-order derivative

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$$

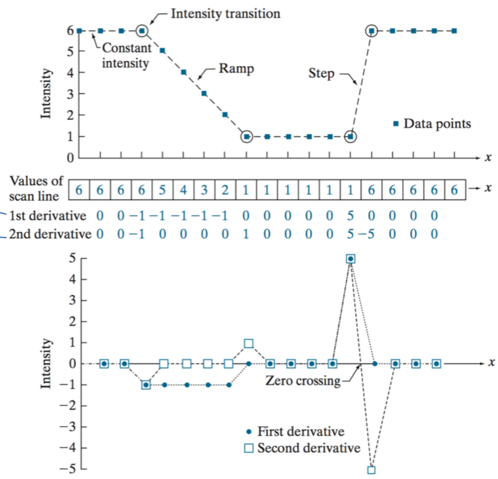


Image credit: Gonzalez, Fig. 3.44

Edge detection using derivatives

(1) Detecting the local maxima or minima of the first derivative

(2) Detecting the zero-crossings of the second derivative

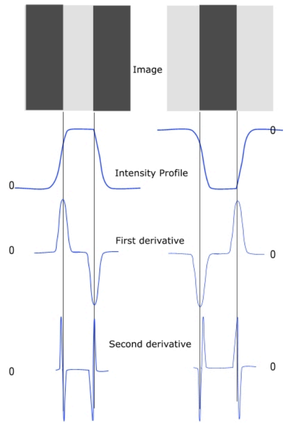


Image credit: MIPAV

Gradient of 2D discrete function

- Gradient of a 2-dimension function is calculated as follows:

$$Grad(x,y) = \frac{\partial f(x,y)}{\partial x}\vec{i} + \frac{\partial f(x,y)}{\partial y}\vec{j}$$

- The gradient is approximated as follows (first-order derivative) :

$$\frac{\partial f(x,y)}{\partial x} = f(x+1,y) - f(x,y), \frac{\partial f(x,y)}{\partial y} = f(x,y+1) - f(x,y)$$

Gradient

- The magnitude of gradient indicates the strong of edges:

$$|Grad(x,y)| = \sqrt{\left(\frac{\partial f(x,y)}{\partial y}\right)^2 + \left(\frac{\partial f(x,y)}{\partial x}\right)^2}$$

- Gradient computation procedure:
  - Calculate column gradient
  - Calculate row gradient
  - Calculate final gradient by the above function

Various kernels used to compute the gradient

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

A 3x3 region of an image

−1	0		0	−1
0	1		1	0

Roberts

$$g_x = \frac{\partial f}{\partial x} = (z_9 - z_5)$$
$$g_y = \frac{\partial f}{\partial y} = (z_8 - z_6)$$

−1	−1	−1		−1	0	1
0	0	0		−1	0	1
1	1	1		−1	0	1

Prewitt

$$g_x = \frac{\partial f}{\partial x} = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$$
$$g_y = \frac{\partial f}{\partial y} = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$$

−1	−2	−1		−1	0	1
0	0	0		−2	0	2
1	2	1		−1	0	1

Sobel

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$
$$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

Pixel Difference masks

Column mask

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$$

Row mask

$$\begin{bmatrix} 0 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Pixel difference example

Original Image

Column edges

Row edges

Final edges

Robert mask

- Roberts masks calculate gradient from two diagonals

Column

$$\begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Row

$$\begin{bmatrix} -1 & & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Robert mask example

Original Image

Column edges

Row edges

Final edges

Prewitt mask

Column	Row
$\frac{1}{3}\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$

Prewitt mask

Original Image

Row edges

Column edges

Final edges

Sobel mask

Column

$$\frac{1}{4}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

Row

$$\frac{1}{4}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

26

Sobel filter example

Original Image

Column

Row

Combination

Laplace gradient

- Laplace edge in a continuous domain

$$G(x,y) = -\left(\frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}\right)$$

- In a discrete domain, Laplace edge is approximated by

$$\begin{aligned} G(x,y) &= [f(x,y) - f(x,y-1] - [f(x,y+1) - f(x,y)] \\ &\quad + [f(x,y) - f(x+1,y] - [f(x-1,y) - f(x,y)] \\ &= f(x,y) * H(x,y) \end{aligned}$$

Laplace mask

$$\begin{aligned} H &= \begin{bmatrix} 0 & 0 & 0 \\ -1 & 2 & -1 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -1 & 0 \\ 0 & 2 & 0 \\ 0 & -1 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \end{aligned}$$

Laplace filter example

Original image I

|I*H|

Highboost filtering with Laplace

- Overall

$$I_{highboost} = c \cdot I_{orginal} + I_{highpass}$$

$$= \left(c \cdot \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + H \right) * I_{original}$$

- Using Laplace mask

$$I_{highboost} = \left(c \cdot \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \right) * I_{original} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & c+4 & -1 \\ 0 & -1 & 0 \end{bmatrix} * I_{original}$$

Highboost filter example

Original Image

c=0.5

c=1

Gradient comparison

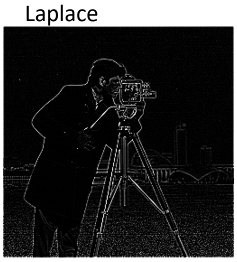
Pixel difference

Robert

Prewitt

Sobel

Laplace



More advanced edge detection

Laplacian of a Gaussian (LoG) (Marr-Hildreth algorithm)

- An edge detection should:
  - Be a differential operator capable of computing a digital approximation of the first or second derivative at every point in the image
  - Be capable of being "tuned" to act at any desired scale, so that large operators can be used to detect blurry edges and small operators to detect sharply focused fine detail

$$\nabla^2 G(x,y) = \left(\frac{x^2 + y^2 - 2\sigma^2}{\sigma^4} \right) e^{-\frac{x^2+y^2}{2\sigma^2}}$$

LoG gradient example

Original image

LoG filter

LoG edge detection

1. Applying LoG to the image
2. Detection of zero-crossings in the image
3. Threshold the zero-crossing to keep only the strong ones (large difference between the positive maximum and the negative minimum)

LoG edge detection example

Original image

LoG filter

LoG edge

Canny edge detection

1. Smooth the input image with a Gaussian filter
2. Compute the gradient magnitude and angle images
3. Apply nonmaxima suppression to the gradient magnitude image
4. Use double thresholding and connectivity analysis to detect and link edges

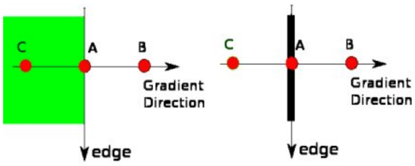
Canny edge detection

1. Smooth with 5x5 Gaussian kernel
2. Gradient with Sobel kernels

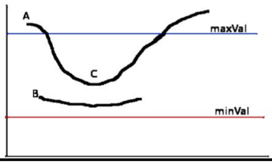
$$Edge\_Gradient\ (G) = \sqrt{G_x^2 + G_y^2}$$

$$Angle\ (\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

3. Non-maximum suppression



4. Thresholding



Canny edge detection example

Original image

Canny

Ideas for Final projects

References

- Student-proposed projects
- https://web.stanford.edu/class/ee368/

Projects Win 2018/19
Projects Win 2017/18
Projects Aut 2016/17
Projects Aut 2015/16
Projects Spr 2014/15
Projects Spr 2013/14
Projects Win 2013/14
Projects Aut 2013/14
Projects Spr 2012/13
Projects Spr 2011/12
Projects Spr 2010/11
Projects Spr 2009/10
Projects Spr 2007/08
Projects Spr 2006/07
Projects Spr 2005/06
Projects Spr 2003/04
Projects Spr 2002/03

Sample projects

- Face detection
- Sudoku solver
- Lottery ticket recognition
- Facial emotion detection
- Vacant parking spot detection
- Money bill recognition
- Cross-word solver
- Road sign detection and distance estimation
- Tic-tac-toe
- Chess board reader
- Image colorization from sketch
- Stereo panorama/rendering
- Monocular-vision based depth estimation