Lecture 17-Edge Detection (Chapter 10.1-10.2.6)

Yuyao Zhang PhD

zhangyy8@shanghaitech.edu.cn

SIST Building 2 302-F

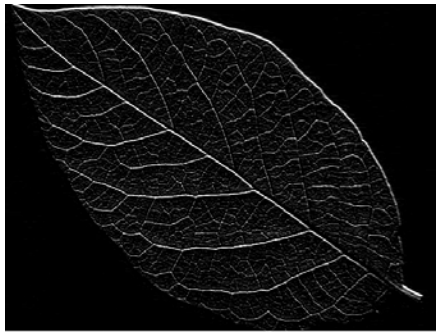
Spatial 2D Filters

Sobel filter

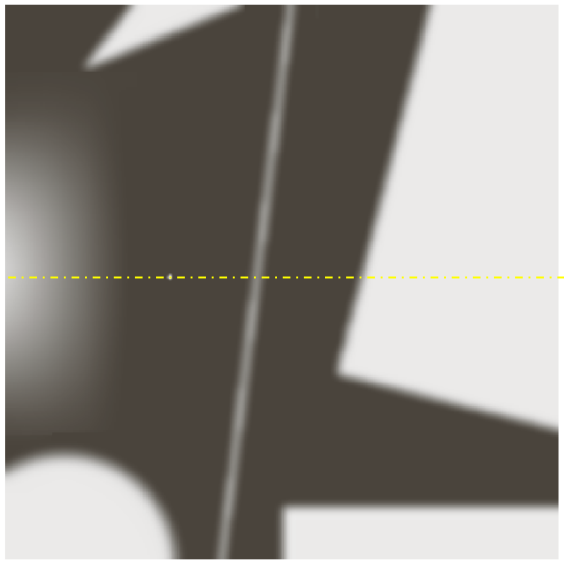
-1	-2	-1
0	0	0
1	2	1

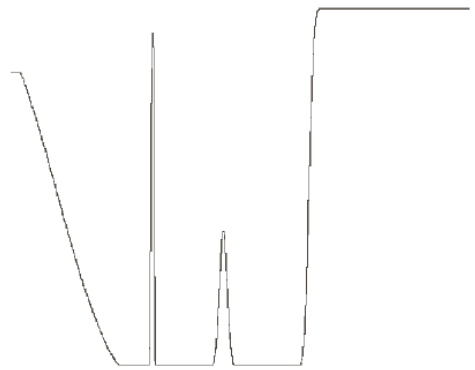
-1	0	1
-2	0	2
-1	0	1

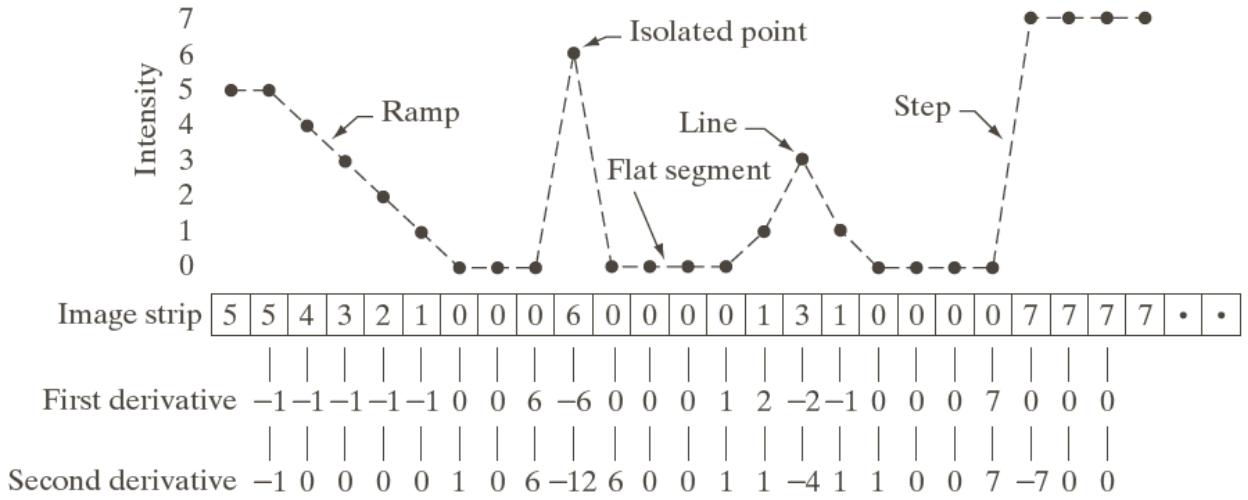
-2	-1	0
-1	0	1
0	1	2



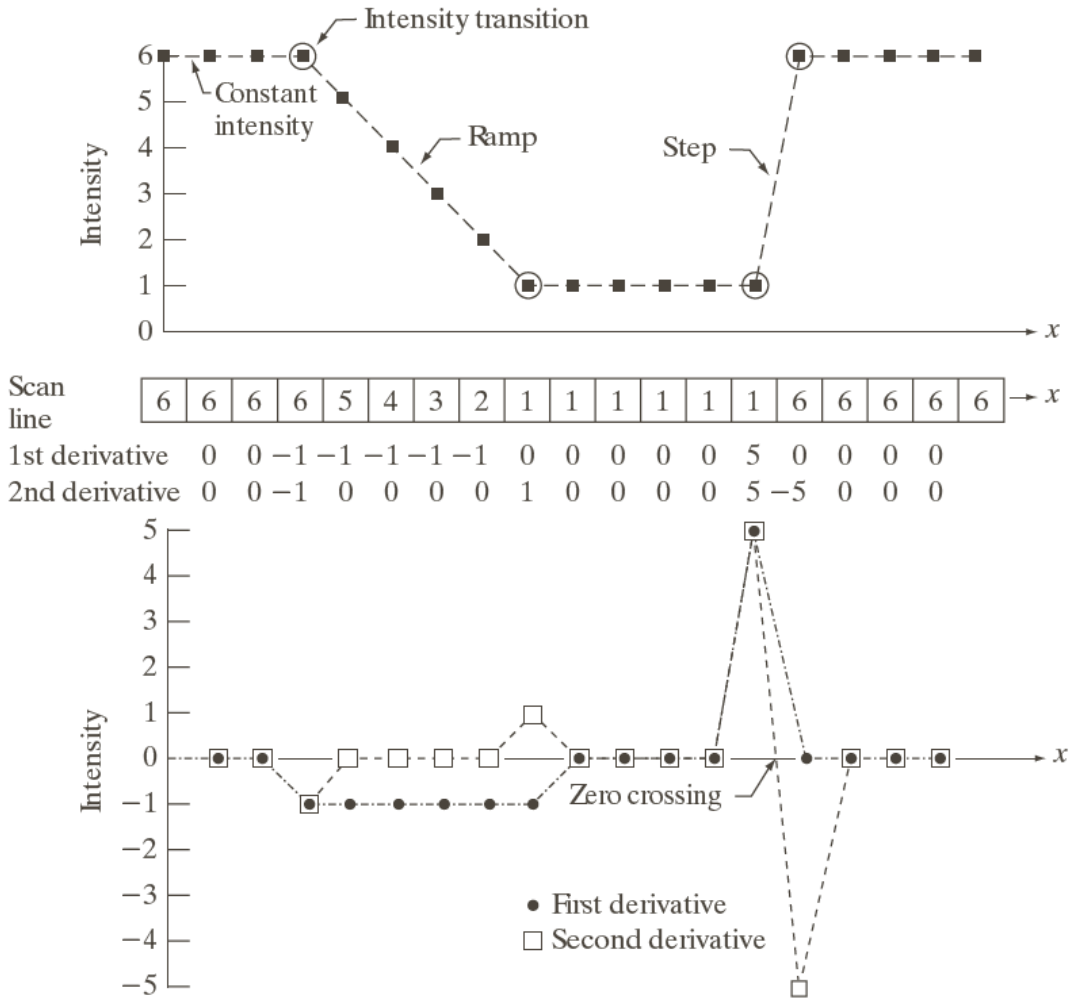
Edge Detection (边缘检测)







Derivatives

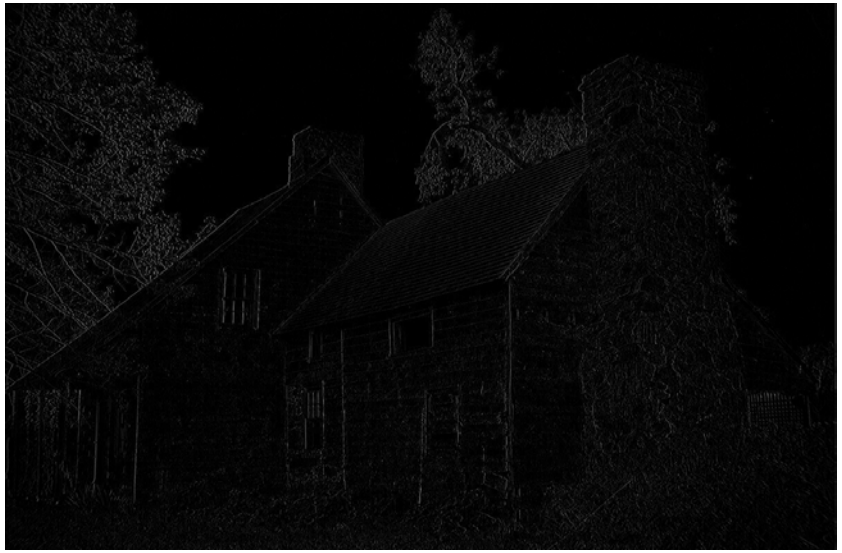
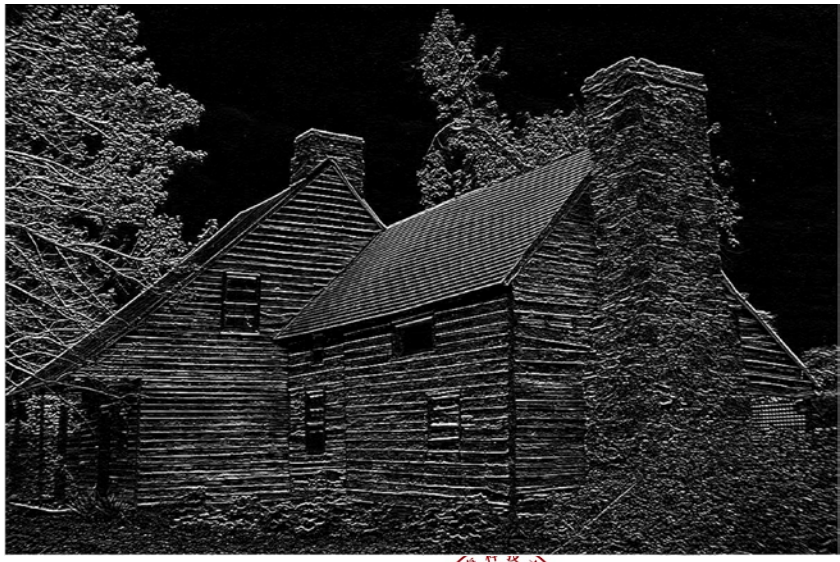


1. Zero in area of constant intensity
2. Nonzero at the onset of intensity step or ramp
3. (1) Nonzero along intensity ramp – 1st order derivative

   (2) Zero along intensity ramp with constant slope – 2nd order derivative

A simple real-world example

- Load image “Peter_Burr_House.jpg”.
- Generate a prewitt/sobel filter for x-direction and a prewitt/sobel for y-direction.
- Apply filters to image and show them.

Gradients

$$\nabla F = grad(F) = \begin{bmatrix} \frac{\partial F}{\partial x} \\ \frac{\partial F}{\partial y} \end{bmatrix} = \begin{bmatrix} g_x \\ g_y \end{bmatrix} \approx \begin{bmatrix} F(x+1, y) - F(x, y) \\ F(x, y+1) - F(x, y) \end{bmatrix}$$

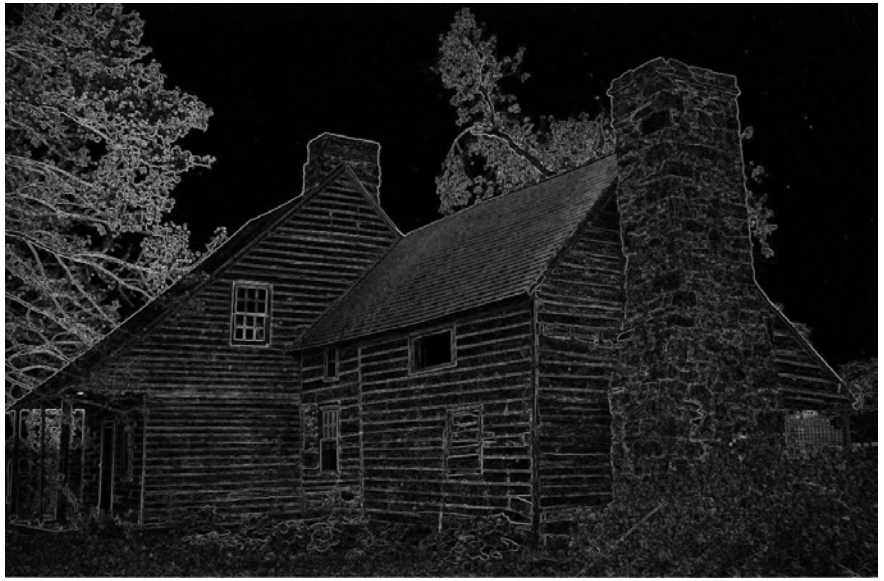
∇F is perpendicular to edge.

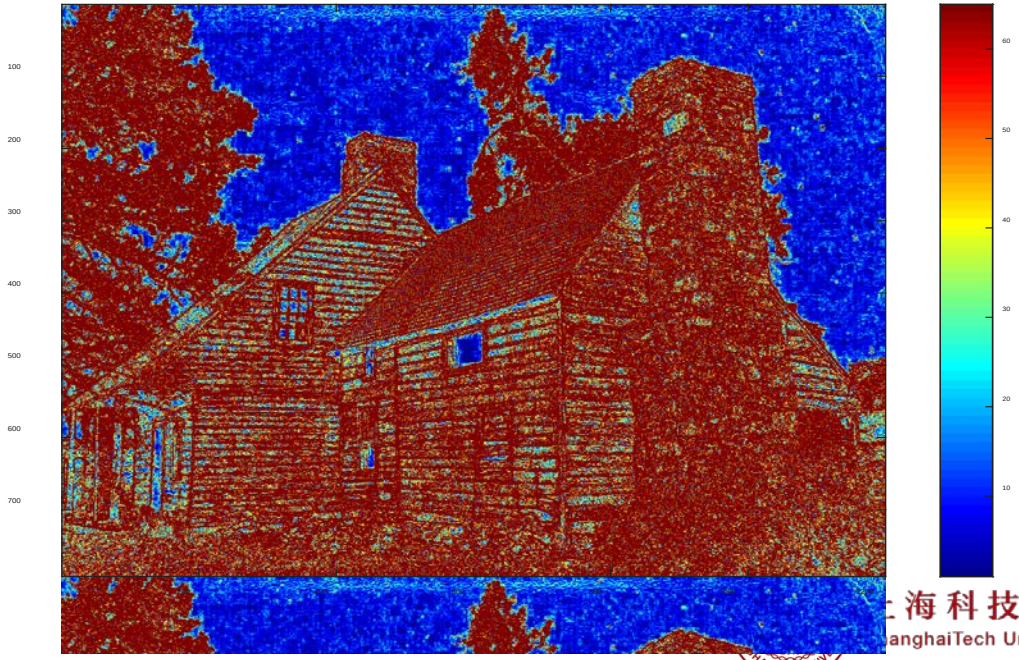
$$M(x, y) = \sqrt{(g_x^2 + g_y^2)}$$

$$\alpha(x, y) = tan^{-1}(g_y / g_x)$$

A simple real-world example

- Load image “Peter_Burr_House.jpg”.
- Generate a sobel/prewitt filter for x-direction and a sobel/prewitt for y-direction.
- Compute the Magnitude and angles of edge gradients.





A simple real-world example

- Load image “Peter_Burr_House.jpg”.
- Generate a sobel/prewitt filter for x-direction and a sobel/prewitt for y-direction.
- Compute the Magnitude and angles of edge gradients.
- Pick out the edges that gradient magnitude is greater than $\tau 1$ and angle is around $\alpha 1$.

```
figure;
edge = (abs(angle+90)<20) & (mag>150);
imshow(edge);
```



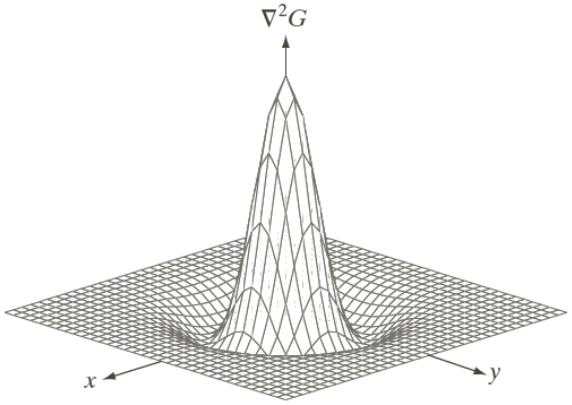
Edge Detectors

- LoG (Laplacian of a Gaussian, 高斯拉普拉斯算子):

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

$$\nabla^2 G(x,y) = \frac{\partial^2 G(x,y)}{\partial^2 x} + \frac{\partial^2 G(x,y)}{\partial^2 y}$$

$$= \left[\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$$



0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

- Try to visualize a LoG:

```
>>h1=fspecial('log',[101,101],3/10/30
);
>>surf(h1,'edgecolor','none');
```

Canny Edge Detectors

- **Canny Detector (坎尼边缘检测器):**

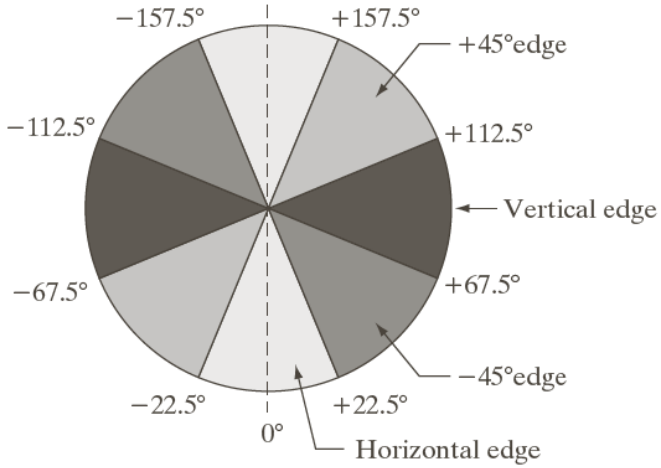
  1. Smooth the input image with a Gaussian filter;
  2. Compute the gradient magnitude and angle images;
  3. Apply nonmaxima suppression (非最大值抑制) to the gradient magnitude image;
  4. Use double thresholding and connectivity analysis to detect and link edge.

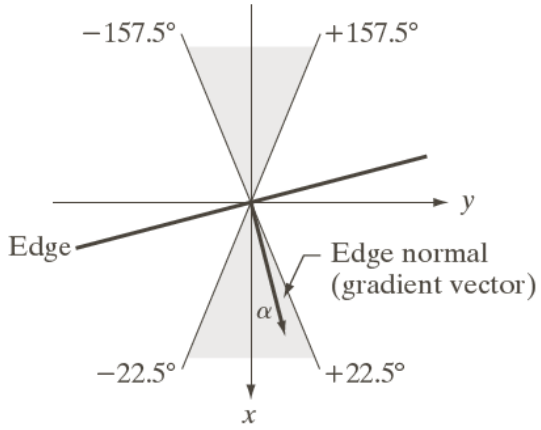
- **Matlab function:** [g, t] = edge(f, 'canny', T, sigma), where T=[T1, T2]

Non-maxima suppression （非最大值抑制）

1. Orientation quantize:

Input： image gradient magnitude and angle map;





2. Non-maxima suppression

If M(x,y) is greater than all its neighbors in the quantized edge direction,

$G_N(x,y) = M(x,y)$, otherwise $G_N(x,y) = 0$.

Double thresholding edge linking

- **Detect and link edges.**

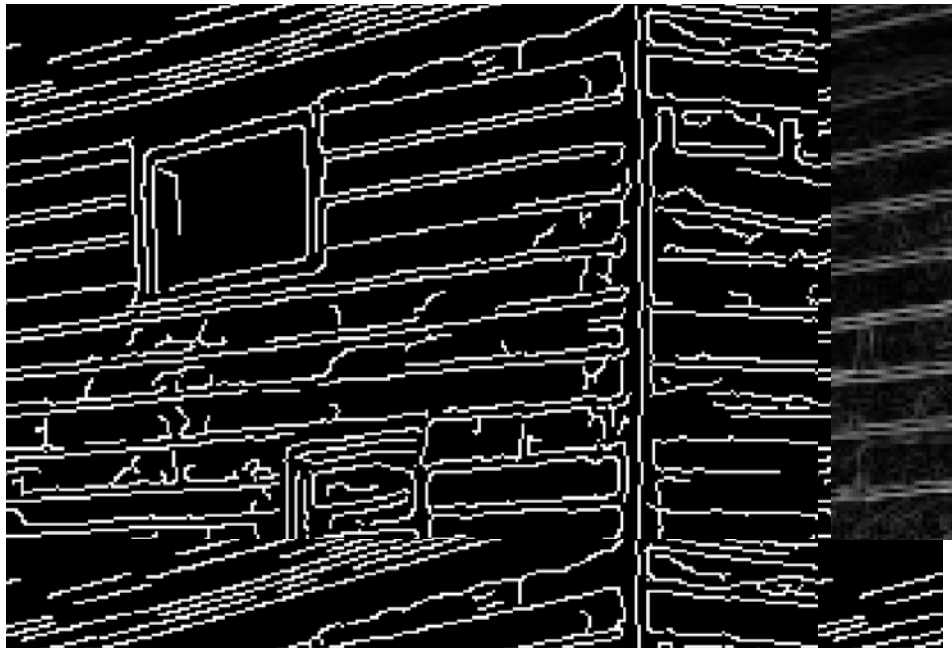
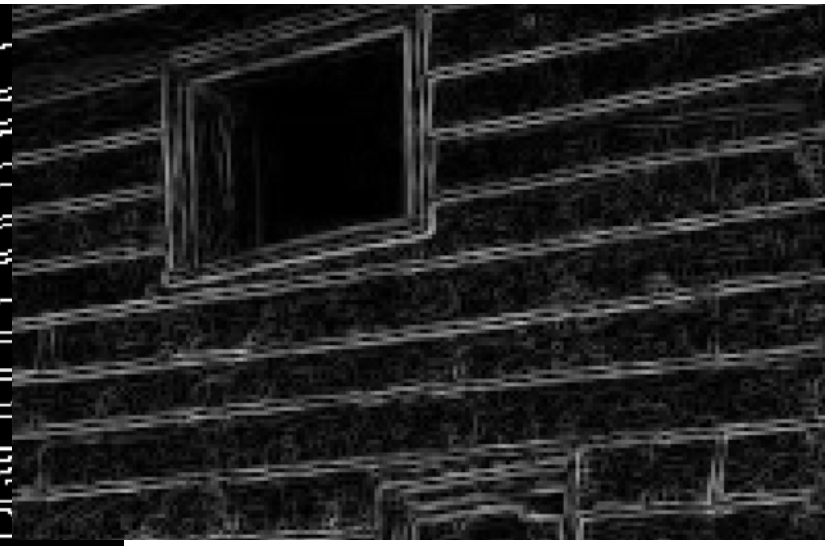
$$G_H(x,y) = G_N(x,y) \geq T_2$$

$$G_L(x,y) = T_1 \leq G_N(x,y) \leq T_2$$

Final map: $G_H(x,y)$ **and** all edges in $G_L(x,y)$ that are adjacent to at least one pixel of $G_H(x,y)$

- **Matlab function:** [g, t] = edge(f, 'canny', T, sigma), where T=[T1, T2]




Take home message

- **Rules:**
  - Seek the 1st derivative greater than a threshold
  - Seek the zero-crossing point on the 2nd derivative
- **Steps:**
  1. Image smoothing for noise reduction
  2. Detection of edge points
  3. Edge localization