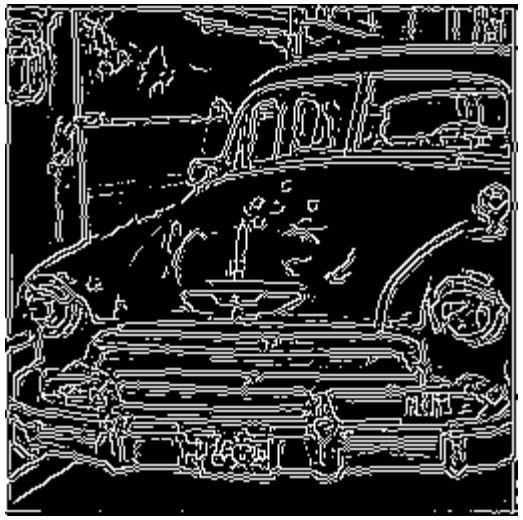
Canny's Edge Detector



Liangliang Cao

Guest Lecture

ECE 547, UIUC

Reference

- John Canny's paper: *A Computational Approach to Edge Detection*, IEEE Trans PAMI, 1986

- Peter Kovesi's Matlab functions

http://www.csse.uwa.edu.au/~pk/Research/MatlabFns/

Part I: Mathematical Model

Goal

- Canny's aim was to discover the ***optimal*** edge detection algorithm:
  - *good detection* – the algorithm should mark as many real edges in the image as possible.
  - *good localization* – edges marked should be as close as possible to the edge in the real image.
  - *minimal response* – a given edge in the image should only be marked once, and where possible, image noise should not create false edges.

Formulation

Detection and Localization criterions

- Impulse response of filter : f(x)
- Edge : G(x)

Assuming edge is centered at x=0, filter's finite response bounded by [-W,W]

The response of the filter to the edge at its center is given by a convolution

$$H_G = \int_{-W}^{+W} G(-x)\, f(x)\, dx$$

Criterion 1: Signal/Noise Ratio

The root mean squared noise will be measured by

$$H_n = n_0 \left[\int_{-W}^{+W} f^2(x)\, dx \right]^{1/2}$$

Now the first criterion , the output signal-to-noise ratio (SNR) is given by

$$\text{SNR} = \frac{\left| \int_{-W}^{+W} G(-x)\, f(x)\, dx \right|}{n_0 \sqrt{\int_{-W}^{+W} f^2(x)\, dx}}$$

Criterion 2: Localization

- For localization we need some measure which increases as localization increases.

$$\text{Localization} = \frac{\left| \int_{-W}^{+W} G'(-x)\, f'(x)\, dx \right|}{n_0 \sqrt{\int_{-W}^{+W} f'^2(x)\, dx}}.$$

Criterion 3: Multiple Response

- The mean distance between adjacent maxima in the output is twice the distance between adjacent zero-crossings in the derivative of the output operator.

$$x_{\max}(f) = 2x_{zc}(f) = kW.$$

$$x_{zc}(f) = \pi \left(\frac{\int_{-\infty}^{+\infty} f'^2(x)\, dx}{\int_{-\infty}^{+\infty} f''^2(x)\, dx} \right)^{1/2}$$

k: the number of noise maxima that could lead to a false response.

General Optimization

It is impossible to directly maximize the SNR, localization under the multiple response constraint.

We use penalty function: non-zero values when one of the constraints is violated

$$SNR(f) * Localization(f) - \sum \mu_i P_i(f)$$

where P_i is a function which has a positive value only when a constraint is violated.

Optimization for special cases

Three kinds of edges:

- Ridge edges

- Roof edges (rare)

- ***Step edges*** (important)

Can be get some analytical solution
for step edges?

Step Edges

- Definition

$$G(x) = Au_{-1}(x) \qquad u_{-1}(x) = \begin{cases} 0, & \text{for } x < 0; \\ 1, & \text{for } x \geq 0; \end{cases}$$

$$\text{SNR} = \frac{A\left|\int_{-W}^{0} f(x)\,dx\right|}{n_0\sqrt{\int_{-W}^{+W} f^2(x)\,dx}} \qquad \text{Localization} = \frac{A|f'(0)|}{n_0\sqrt{\int_{-W}^{+W} f'^2(x)\,dx}}.$$

Optimization for Step Edges

The problem can be solved by variational method

Filter Parameters						
n	x_{max}	$\Sigma\Lambda$	r	α	ω	β
1	0.15	4.21	0.215	24.59550	0.12250	63.97566
2	0.3	2.87	0.313	12.47120	0.38284	31.26860
3	0.5	2.13	0.417	7.85869	2.62856	18.28800
4	0.8	1.57	0.515	5.06500	2.56770	11.06100
5	1.0	1.33	0.561	3.45580	0.07161	4.80684
6	1.2	1.12	0.576	2.05220	1.56939	2.91540
7	1.4	0.75	0.484	0.00297	3.50350	7.47700

where filter 6 can be approximate by the first derivative of Gaussian!

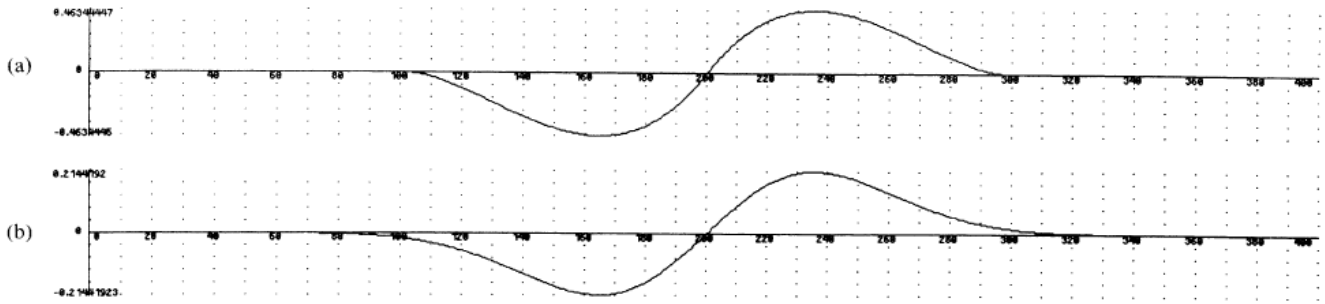


Fig. 6. (a) The optimal step edge operator. (b) The first derivative of a Gaussian.

Summarization

- The type of linear operator that provides the best compromise between noise immunity and localization is ***First Derivative of Gaussian***

- This operator corresponds to smoothing an image with Gaussian function and then computing the gradient

Part II: Implementation

Overview of Procedure

1) Smooth image with a Gaussian

2) Compute the Gradient magnitude using approximations of partial derivatives

3) Thin edges by applying non-maxima suppression to the gradient magnitude

4) Detect edges by double thresholding

Noise Reduction

First the image is convolved with a Gaussian filter

- General rule for both edge and corner detection: robust to noise

Code:

```
sze = ceil(6*sigma);
if ~mod(sze,2)
        sze = sze+1;
end
h = fspecial('gaussian', [sze sze], sigma);
im = filter2(h, im);
```

Gradient

Find the first derivative in horizontal and vertical direction.

$$G_x = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \qquad G_y = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}$$

(note: other more complicated filter can be used)

From this the edge gradient and direction

$$M[i,j] = \sqrt{P[i,j]^2 + Q[i,j]^2}$$

$$\theta[i,j] = \tan^{-1}(Q[i,j], P[i,j])$$

The edge direction angle is rounded to one of four angles representing vertical, horizontal and the two diagonals (0, 45, 90 and 1 degrees for example).

Gradient (code)

- A naïve implementation by Matlab

```
[gx,gy] = gradient(im);
```

- A better implementation by Peter

```
p = [0.037659  0.249153  0.426375  0.249153  0.037659];
d1 =[0.109604  0.276691  0.000000 -0.276691 -0.109604];
gx = conv2(p, d1, im, 'same');
gy= conv2(d1,p,im,'same');
```

- Compute gradient and direction

```
gradient = sqrt(gx.*gx + gy.*gy);
or = atan2(-gy, gx);
or(or<0) = or(or<0)+pi;          % Map angles to 0-pi.
or = or*180/pi;                  % Convert to degrees.
```

Non-maximum suppression

Does the gradient magnitude correspond a local maximum in the gradient direction?

The answer depends on the round angle:

- [angle=0]: the edge point's intensity is greater than the intensities in the **west and east** directions,
- [angle=90]: the edge point's intensity is greater than the intensities in the **north and south** directions,
- [angle=135]: the edge's intensity is greater than the intensities in the **north west and south east** directions,
- [angle=45]: the edge's intensity is greater than the intensities in the **north east and south west** directions.

Edge Thinning

Remove the unwanted spurious points

- Use morphological operations

Code

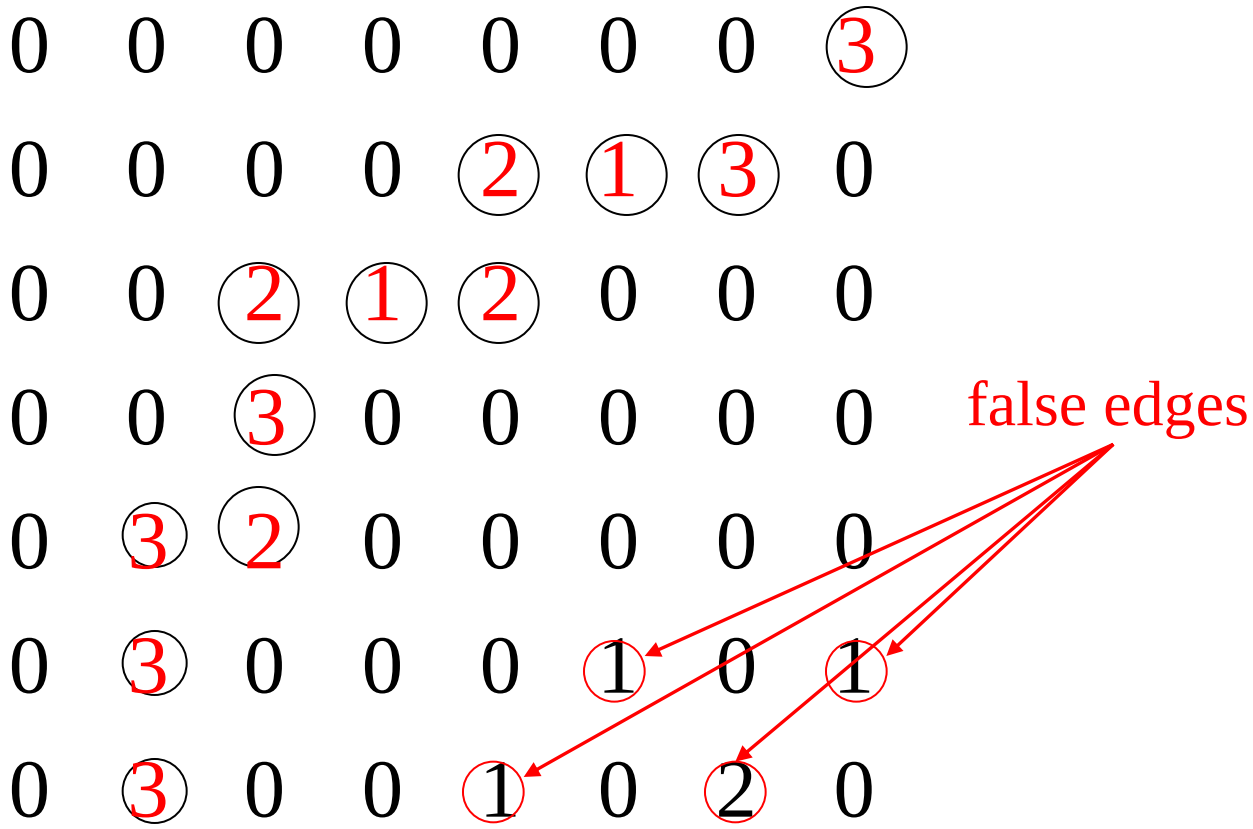
```
skel = bwmorph(im,'skel',Inf);
im = im.*skel;
location = location.*skel
```

Example

0	0	0	0	1	1	1	3
0	0	0	1	2	1	3	1
0	0	2	1	2	1	1	0
0	1	3	2	1	1	0	0
0	3	2	1	0	0	1	0
2	3	2	0	0	1	0	1
2	3	2	0	1	0	2	1

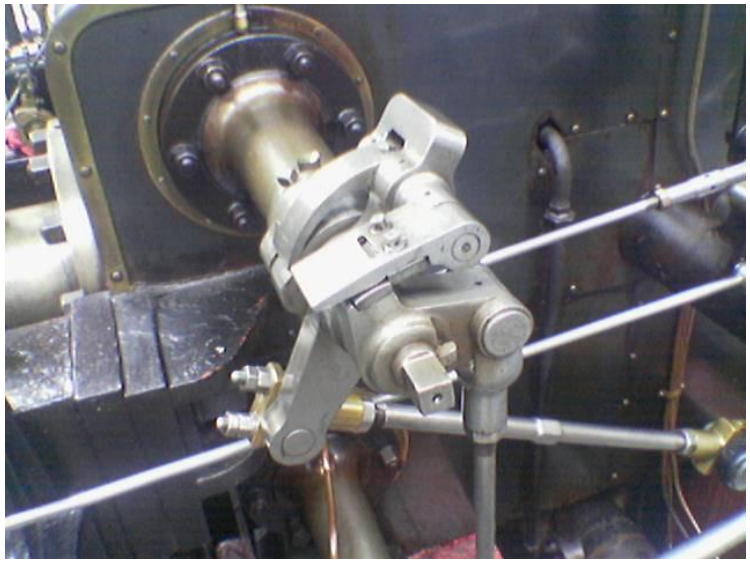
Large gradient are potential edge points

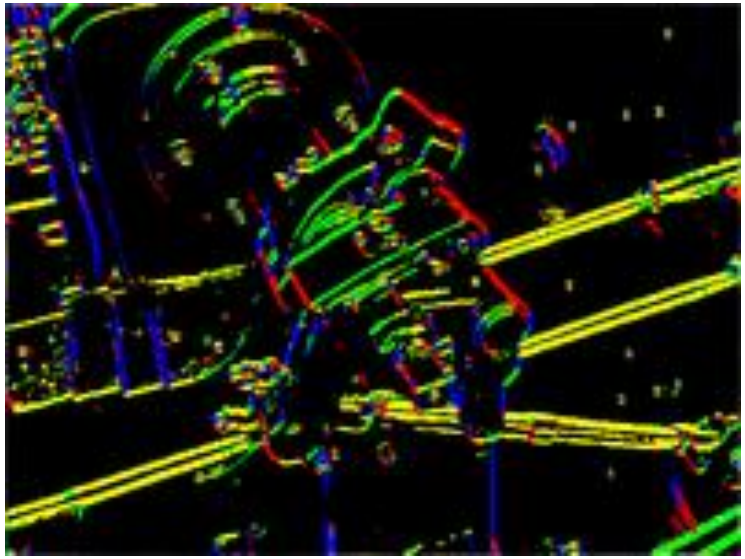
Example



Remove those gradient that are only one pixel wide by edge thinning

Real Example





0 ° 90 °

45 ° 135 °

Tracing Edges Through Hysteresis

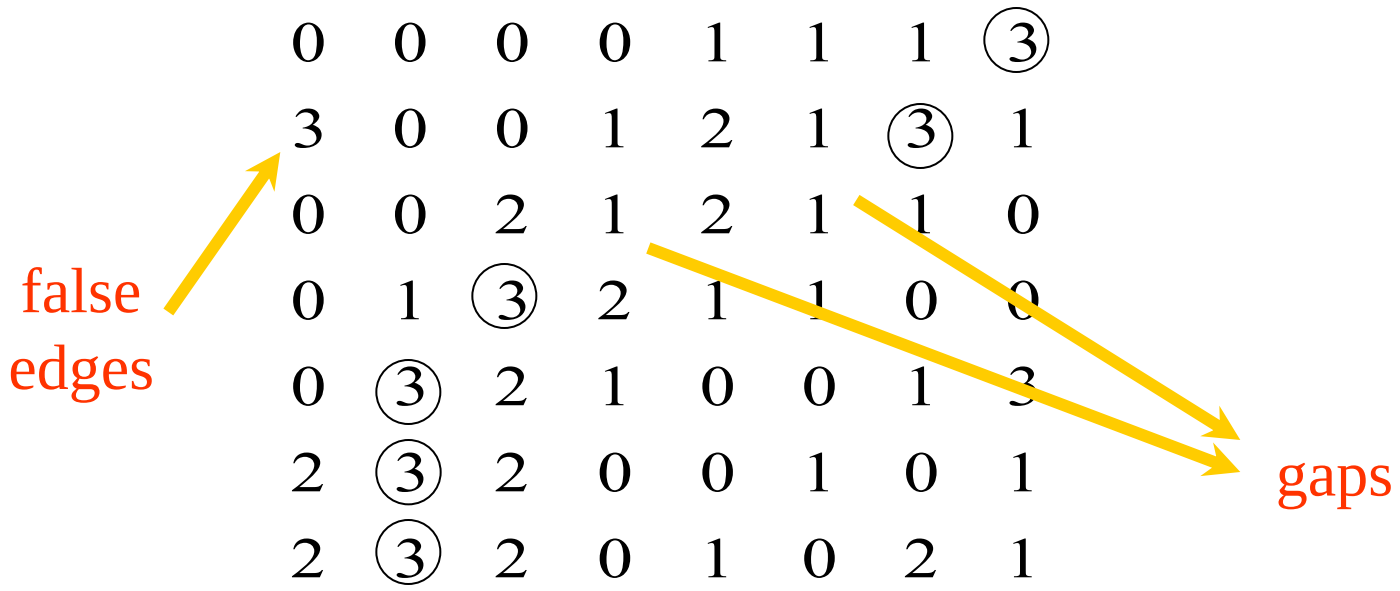
Two thresholds T_h and T_l

- Keep those higher than T_h
- Remove those lower than T_l
- For gradient between T_l and T_h , only kept if they form a continuous edge line with pixels with high gradient magnitude

Code

```
aboveTl = (im > Tl);
[aboveThr, aboveThc] = find(im > Th);
% Obtain all connected regions in above T_l that include a point
% that has a value above T_h
bw = bwselect(aboveTl, aboveThc, aboveThr, 8);
```

Large Value of Gradient

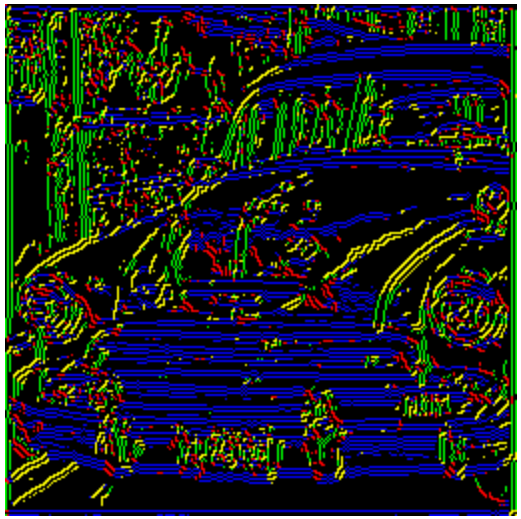


suppressing all values along the line of the gradient that are not peak values

Parameters

1. The size of the Gaussian filter:
   - Smaller filters cause less blurring, and allow detection of small, sharp lines.
   - A larger filter causes more blurring, smearing out the value of a given pixel over a larger area of the image. Larger blurring radii are more useful for detecting larger, smoother edges (e.g., the edge of a rainbow)
2. High and low thresholds in hysteresis: allows more flexibility than in a single-threshold approach
   - A threshold set too high can miss important information.
   - A threshold set too low will falsely identify irrelevant information (such as noise) as important.

Real Example



Conclusion

- Canny edge detector has been still arguably the best edge detector for the last twenty years
- The operator of Gradient of Gaussian has rich theoretical meaning
- Beyond edges, corner detector is more popular in recent image recognition
  - SIFT talked by Mert Dikmen