CSE 185

Canny Edge Detector

- State-of-the-art method for performing edge detection.
- Output is binary image

  0 - no edge

  1 - edge

- Exploits
  - whether pixel is in "direction of edge"
  - whether pixel is adjacent to a "strong edge"

Objectives:

1) Low error rate. All edges should be found, and there should be no spurious edges.

2) Edge points should be well localized. The edges located must be as close as possible to the true edges. That is, the distance between a point marked as an edge by the detector and the center of the true edge should be minimal.

3) Single edge point response. The detector should return only one point for each true edge point. That is, the number of local maxima around the true edge should be minimal. This means that the detector should not identify multiple edges where only a single edge point exists.

Canny's contribution was expressing these criteria mathematically and then attempting to find optimal solutions.

Steps:
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.

1. Smooth the image with a Gaussian filter.

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

Form smoothed image by convolving G and f:

$$f_s(x,y) = G(x,y) * f(x,y)$$

2. Compute gradient magnitude and angle images.

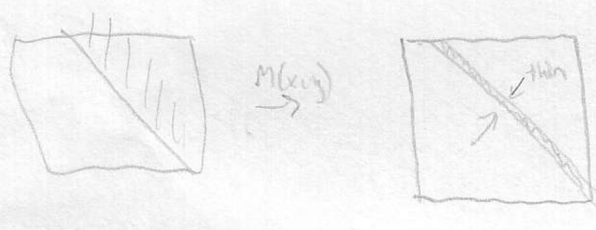
$$M(x,y) = \sqrt{g_x^2 + g_y^2}$$

$$\alpha(x,y) = \tan^{-1}\left[\frac{g_y}{g_x}\right]$$

3. Apply nonmaxima suppression to the gradient magnitude image.

$M(x,y)$ typically contains wide ridges around local maxima.

Use nonmaxima suppression to thin those edges



a. Quantize edge directions into 4 bins:

$-22.5° < \alpha(x,y) \leq 22.5° \rightarrow$ horizontal
$67.5° < \alpha(x,y) \leq 112.5° \rightarrow$ vertical
$112.5° < \alpha(x,y) \leq 157.5° \rightarrow 45°$
$+22.5° < \alpha(x,y) \leq 67.50 \rightarrow -45°$

If $\alpha(x,y) > 157.5$ then $\alpha(x,y) = \alpha(x,y) - 180°$

See Figure 10.24c

Find direction $d_k = \{hor, vert, 45°, -45°\}$ that is closest to $\alpha(x,y)$

b. If the value of $M(x,y)$ is less than at least one of its two neighbors along d_k, let $g_N(x,y) = 0$ (suppression); otherwise, let $g_N(x,y) = M(x,y)$.

See Figure 10.24 a, b

Image $g_N(x,y)$ contains only the thinned edges; it is equal to $M(x,y)$ with the nonmaxima edge points suppressed.

4. Use double thresholding and connectivity analysis to detect and link edges.

Final step is to threshold $g_N(x,y)$ to reduce false edge points.

Problem with single threshold

- Value too low → still get some false edges (false positives)
- Value too high → valid edge points will be eliminated (false negatives)

Instead, hysteresis thresholding which uses two thresholds T_L and T_H.

Create two additional images

$$g_{NH}(x,y) = g_N(x,y) \geq T_H \qquad \text{"strong edges"}$$

$$g_{NL}(x,y) = g_N(x,y) \geq T_L \qquad \text{"strong + weak edges"}$$

where g_{NH}, g_{NL} are otherwise zero.

Remove strong edges from g_{NL}:

$$g_{NL}(x,y) = g_{NL}(x,y) - g_{NH}(x,y) \qquad \text{"weak edges"}$$

All non-zero pixels in g_{NH} are marked as edges.

To try reduce gaps in edges in g_{NH}, do the following:

(a) Locate the next unvisited edge pixel, p, in $g_{NH}(x,y)$.
(b) Mark as valid edge pixels all the weak pixels in in $g_{NL}(x,y)$ that are connected to p using, say, 8-connectivity.
(c) If all nonzero pixels in $g_{NH}(x,y)$ have been visited go to step d.
Else, return to step a.
(d) Set to zero all pixels in $g_{NL}(x,y)$ that were not marked as valid edge pixels.

Finally merge $g_{NH}(x,y)$ and $g_{NL}(x,y)$ (through OR for example) to form final binary output.

④ Example.

$g_N(x,y)$

200				
	100			
		100		
110			100	
				200

$T_L = 50 \qquad T_H = 150$

$g_{NL}(x,y)$

	100			
		100		
110			100	

$g_{NH}(x,y)$

200				
				200

Result

1				
	1			
		1		
			1	
				1

Figures 10.25 and 10.26

Matlab scripts.