Canny Edge operator

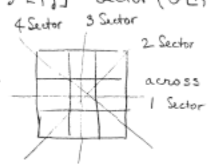
1. Smooth image with a Gaussian filter
2. Compute gradient magnitude and direction
3. Apply non-maximal suppression to the gradient magnitude
4. Use double thresholding to detect and link edges

Non-Maximal Suppression

- M[i,j] will have large values where gradient is large. We still need to find local maxima in this array to locate edges.
- Must <u>thin</u> so only points of greatest local change remain.

Non-Maximal Suppression

① reduce angle θ to one of four sectors. $\zeta[i,j] = \text{Sector}(\theta[i,j])$.

② pass 

across $m[i,j]$. This is a gradient image

③ compare m to the two neighbors in direction specified by $\zeta[i,j]$.

④ If $m[i,j]$ not larger, then $m[i,j] = 0$.

→ denote this entire process $N[i,j] = nms(m[i,j], \zeta[i,j])$

Double Thresholding

- After non-maximal suppression image contains many false edge fragments caused by noise and fine texture
- Threshold N[i,j], but good results are difficult to achieve with a single threshold T.
- Use two thresholds T_1 and T_2. Initially link contours using threshold T_1. If a gap is encountered drop to threshold T_2 until you rejoin a T_1 contour.

Canny operator example

(http://homepages.inf.ed.ac.uk/rbf/HIPR2/canny.htm)

(a) Original image

(b) Canny, σ=1.0, T_1=255, T_2=1

Using a Gaussian kernel with standard deviation 1.0 and upper and lower thresholds of 255 and 1

Most of the major edges are detected and lots of details have been picked out well --- note that this may be too much detail for subsequent processing. The `Y-Junction effect' mentioned above can be seen at the bottom left corner of the mirror.

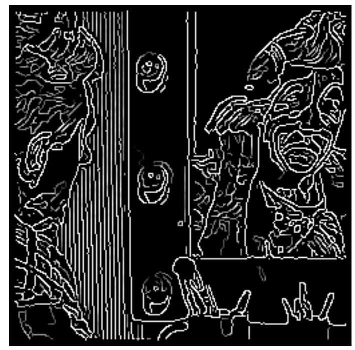
(a) Original

(b) Canny, σ=1.0, T_1=255, T_2=220

Using a Gaussian kernel with standard deviation 1.0 and upper and lower thresholds of 255 and 220.

The image is obtained using the same kernel size and upper threshold, but with the lower threshold increased to 220. The edges have become more broken up than in the previous image, which is likely to be bad for subsequent processing. Also, the vertical edges on the wall have not been detected, along their full length.

(a) Original



(b) Canny, σ=1.0, T_1=128, T_2=1

Using a Gaussian kernel with standard deviation 1.0 and upper and lower thresholds of 255 and 1

The image is obtained by lowering the upper threshold to 128. The lower threshold is kept at 1 and the Gaussian standard deviation remains at 1.0. Many more faint edges are detected along with some short `noisy' fragments. Notice that the detail in the clown's hair is now picked out.

(a) Original

(b) Canny, σ=2.0, T_1=128, T_2=1

Using a Gaussian kernel with standard deviation 1.0 and upper and lower thresholds of 255 and 1

The image is obtained with the same thresholds as the previous image, but the Gaussian used has a standard deviation of 2.0. Much of the detail on the wall is no longer detected, but most of the strong edges remain. The edges also tend to be smoother and less noisy.

Edge Relaxation

Improve edge operator estimate by re-adjusting edge estimate based upon local information

algorithm:

of any edge

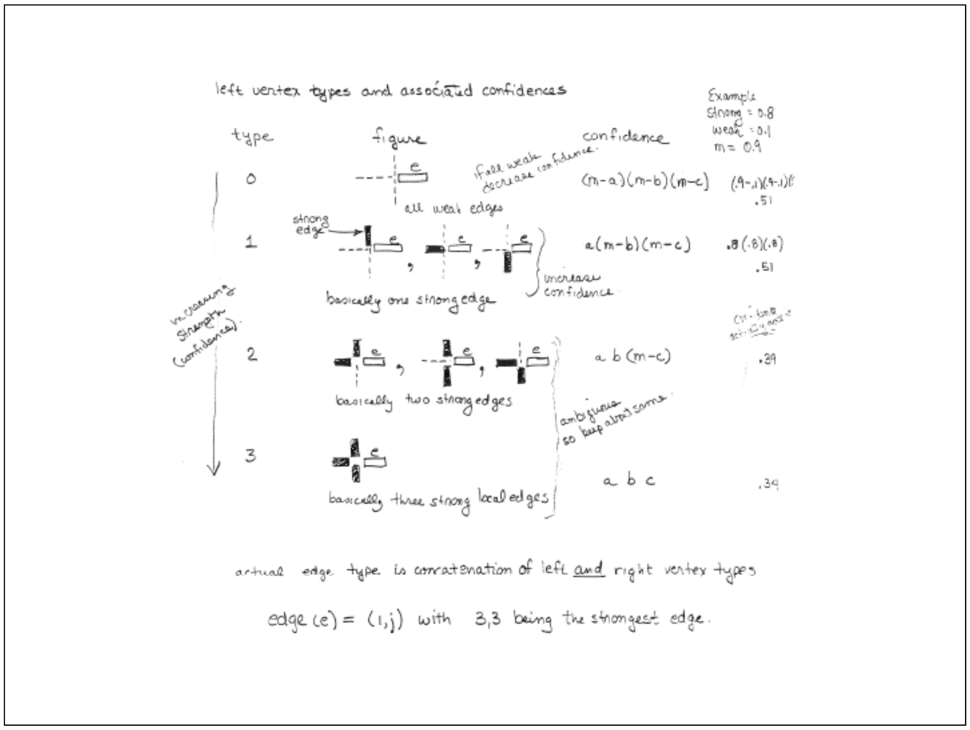
$$C^0(e) = \frac{\text{gradient magnitude}}{\text{maximum gradient amplitude in image}}$$

$k = 1$

initial confidence of edge is normalized gradient

while any $C^k(e) \neq (0 \text{ or } 1)$ do
begin
classify edge: edge_type = f(edge_neighbors)
adjust confidence: $C^k(e) = f(\text{edge\_type}, C^{k-1}(e))$
iteration counter: $k = k + 1$
end

algorithm forces all points in edge image to converge to 0 or 1


left vertex types and associated confidences
Example
strong = 0.8
weak = 0.1
m = 0.9
type
figure
confidence
if all weak decrease confidence.
0
e
all weak edges
(m-a)(m-b)(m-c)
.51
1
strong edge
basically one strong edge
increase confidence
a(m-b)(m-c)
.8(.8)(.8)
.51
increasing strength (confidence)
2
basically two strong edges
a b (m-c)
.39
ambiguous so keep about same
3
basically three strong local edges
a b c
.39
actual edge type is concatenation of left and right vertex types
edge (e) = (i,j) with 3,3 being the strongest edge.

Edge Relaxation Example

(Source: Ballard & Brown)

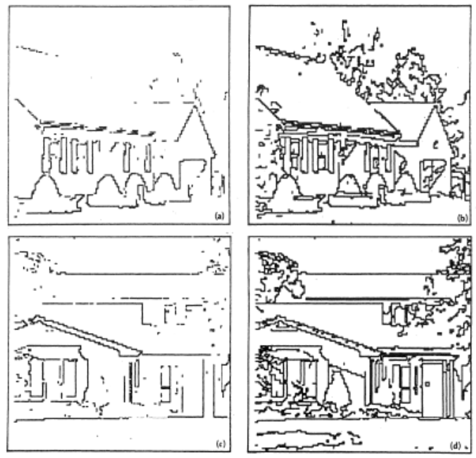
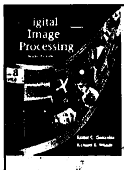


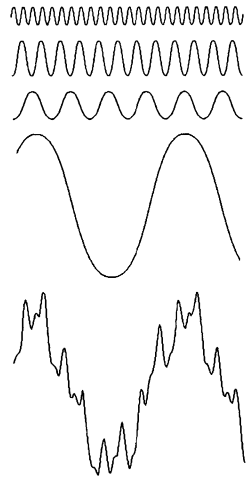
Fig. 3.22 Edge relaxation results. (a) Raw edge data. Edge strengths have been thresholded at 0.25 for display purposes only. (b) Results after five iterations of relaxation applied to (a). (c) Different version of (a). Edge strengths have been thresholded at 0.25 for display purposes only. (d) Results after five iterations of relaxation applied to (c).



Digital Image Processing, 2nd ed.

www.imageprocessingbook.com

Chapter 4
Image Enhancement in the Frequency Domain



sum of above four functions gives this noisy looking signal.

FIGURE 4.1 The function at the bottom is the sum of the four functions above it. Fourier's idea in 1807 that periodic functions could be represented as a weighted sum of sines and cosines was met with skepticism.

1-D Fourier transform:

$$F(u) = \int_{-\infty}^{\infty} f(x)\, e^{-j 2\pi u x}\, dx$$

$$f(x) = \int_{-\infty}^{+\infty} F(u)\, e^{j 2\pi u x}\, du$$

2-D Fourier transform

$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{+\infty} f(x,y)\, e^{-j2\pi(ux+vy)}\, dx\, dy$$

$$f(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} F(u,v)\, e^{+j2\pi(ux+vy)}\, du\, dv$$

Discrete Fourier Transform

$$F(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(x)\, e^{-j\frac{2\pi u x}{M}} \qquad u = 0, 1, \dots, M-1$$

$$f(x) = \sum_{u=0}^{M-1} F(u)\, e^{j\frac{2\pi u x}{M}} \qquad x = 0, 1, \dots, M-1$$

GWE 4.1

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)\, e^{-j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

where u,v are frequency variables

$$f(x,y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \underbrace{F(u,v)}_{\text{Fourier coefficients}}\, e^{+j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

If $f(x,y)$ is real $F(u,v) = F^*(-u,-v)$

$$\Rightarrow |F(u,v)| = |F(-u,-v)|$$

By substitution

$$F(u,v) = F(u+M,v) = F(u,v+N) = F(u+M,v+N)$$

$\Rightarrow$ infinitely periodic in u & v

From the inverse DFT $f(x,y)$ is periodic in x and y.

$$f(x,y) = f(x+M,y) = f(x,y+N) = f(x+M,v+N)$$

Some properties of the 2-D Fourier Transform

Translation

$$f(x,y)\, e^{j2\pi\left(\frac{u_0x}{M} + \frac{v_0y}{N}\right)} \Leftrightarrow F(u-u_0, v-v_0)$$

$$f(x-x_0, y-y_0) \Leftrightarrow F(u,v)\, e^{-j2\pi\left(\frac{u_0x}{M} + \frac{v_0y}{N}\right)}$$

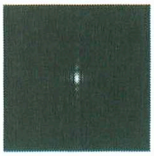
incidentally if we put $u_0 = \frac{M}{2}$ and $v_0 = \frac{N}{2}$ we get

$F(u-\frac{M}{2}, v-\frac{N}{2})$ and

$$e^{j2\pi\left(\frac{u_0x}{M} + \frac{v_0y}{N}\right)} = e^{j2\pi\left(\frac{x}{2} + \frac{y}{2}\right)} = e^{j\pi(x+y)} = (-1)^{x+y}$$

MATLAB/Image Processing Toolbox

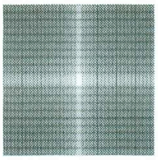
MATLAB Fourier transforms



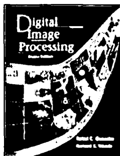
```
>> f=imread('Figure_Rectangle.jpg');  % load in spatial rectangle
>> F=fft2(f);                         % do 2D FFT
>> S=abs(F);                          % determine magnitude for disp
>> imshow(S, [ ])                     % shows in four corners of displa
>> Fc=fftshift(F);                    % shift FFT to center
>> imshow(abs(Fc), [ ]);              % show magnitude of FFT in cen

% much tougher to do display transform
>> g=im2unit8(mat2gray(log(1+double(f))));
>> imshow(g)
% double converts the image to double precision floating point
% mat2gray brings the values to the range [0,1]
% im2unit8 brings the values back to the range [0,255]

% general log transform
>> g=im2unit8(mat2gray(c*log(1+double(f))));
```



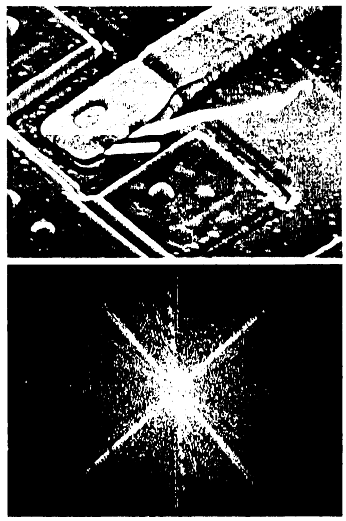
SEE GWE, Section 4.2 Computing and Visualizing the 2-D DFT in MATLAB
GWE, Section 3.2.2 Logarithmic and Contrast Stretching Transformations



Digital Image Processing, 2nd ed.

www.imageprocessingbook.com

Chapter 4
Image Enhancement in the Frequency Domain



a
b

FIGURE 4.4 (a) SEM image of a damaged integrated circuit. (b) Fourier spectrum of (a). (Original image courtesy of Dr. J. M. Hudak, Brockhouse Institute for Materials Research, McMaster University, Hamilton, Ontario, Canada.)

due to white oxide extrusions

value $F(0,0) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)$ which is the average image intensity

Note that $|F(u,v)| = |F(-u,-v)|$

In this figure the strong axes of $|F(u,v)|$ correspond to the approximate $\pm 45°$ edges in the SEM image.

The angle of the white oxide extrusions wrt horizontal roughly corresponds to the short white off-center line in the transform.