MATLAB을 이용한

디지털 영상처리의 기초

9.1 서 론

영역분할은 영상을 부분으로 구분하거나 물체를 분리하는 처리이다. 이 장에서 매우 중요한 2가지의 토픽(topic), 즉 문턱치 처리 및 에지검출을 다룬다.

9.2 문턱치 처리(Thresholding)

9.2.1 단일 문턱치 처리

그레이 영상을 먼저 원 영상에서 그레이레벨 T를 선택하여 이진(흑색과 백색)영상으로 변환하고, 모든 화소에 대하여 T보다 작으면 흑색으로, T보다 크면 백색으로 아래와 같이 치환한다.

$$\text{A pixel becomes } \begin{cases} \text{white if its gray level is} > T, \\ \text{black if its gray level is} \le T. \end{cases}$$

영상에서 특정 물체를 분리하는 것처럼 기본적인 영역분할 문제에서 문턱치 처리가 중요할 뿐만 아니라 컴퓨터(로봇)비젼에서도 매우 중요한 부분이다.

문턱치 처리는 MATLAB에서 아주 간단히 처리할 수 있다. 변수 X로서 저장된 8bit 영상을 생각하자. 아래의 명령으로 간단히 처리할 수 있다.

X > T

imshow로 이 결과를 볼 수 있다. 예를 들면 아래 명령은 그림 9.1과 같은 결과를 얻을 수 있다.

```
>> r=imread('rice.tif');
>> imshow(r),figure,imshow(r>110)
```

주어진 영상 : r

110 보다 클 경우 백색

결과 영상은 이후에 쌀의 개수나 평균 사이즈를 구하는데 이용할 수 있다.

실행 방법을 알기 위해 MATLAB에서 매트릭스를 적용할 때 단일 값의 적용은 매트릭스의 모든 요소에 동시에 적용하며, 이것을 벡터화라 부르는데 부록 A에서 설명한다. 명령 X〉T이면, 그레이 값들이 T보다 크면 1(참)로 나타내고 T보다 작거나 같은 값을 가지는 모든 화소들의 값을 0으로 나타낸다. 이렇게 해서 0과 1로 구성되는 매트릭스로 표현되어 이진 영상으로 볼 수 있다.

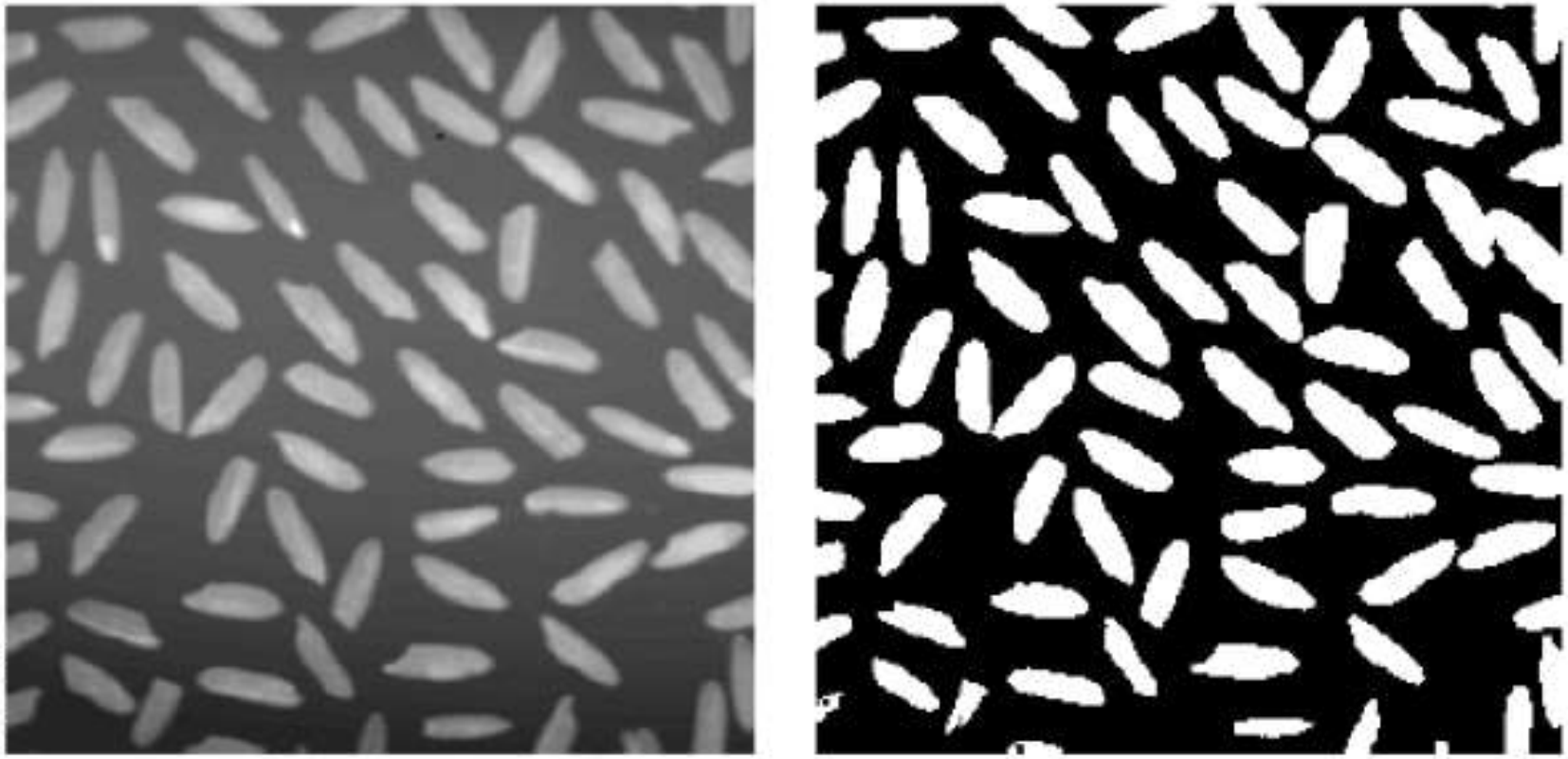


그림 9.1 문턱치 처리에 의한 쌀의 영상

그림 9.1의 쌀 영상은 어두운 배경에 쌀은 밝게 보인다. 밝은 배경에 어두운 물체의 영상도 같게 처리될 수 있으며 아래와 같이 처리하면 그림 9.2와 같은 영상을 얻을 수 있다.

```
>> b=imread('bacteria.tif');
>> imshow(b),figure,imshow(b>100)
```

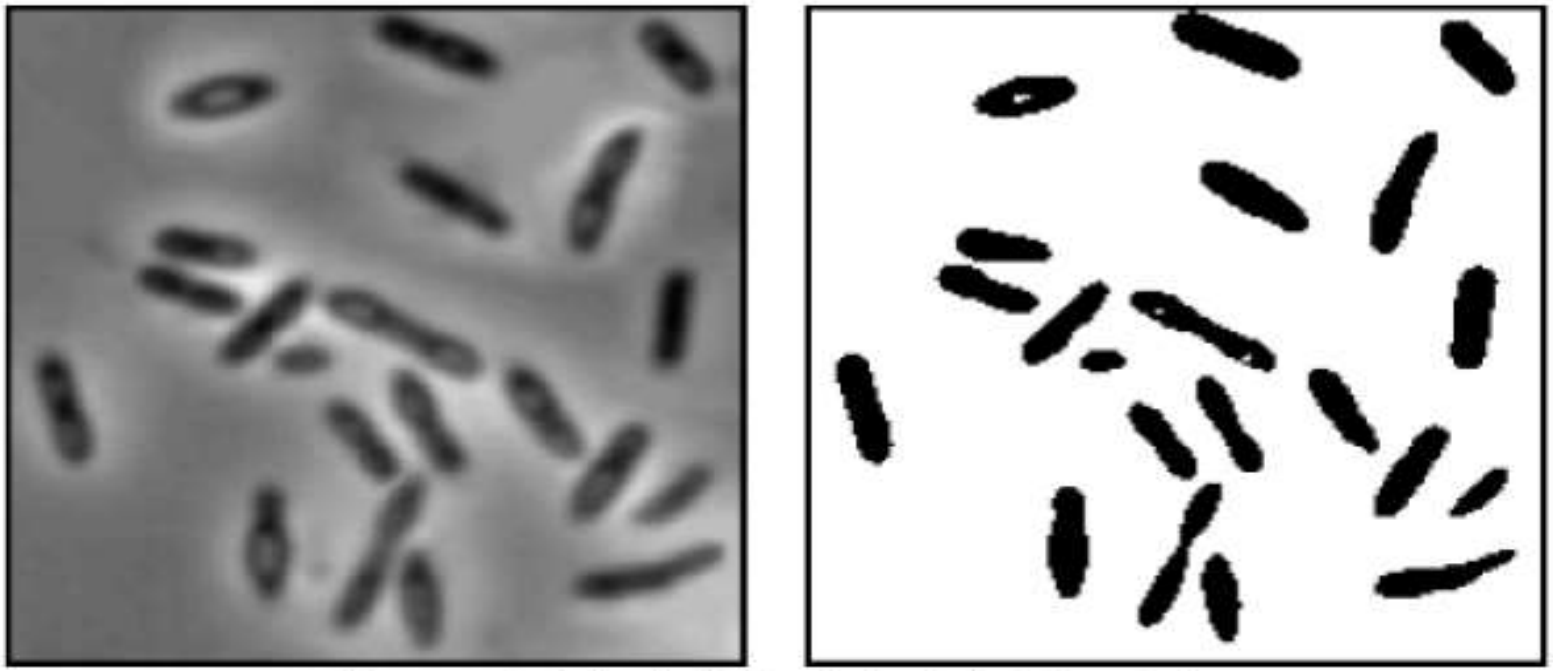


그림 9.2 문턱치 처리에 의한 박테리아 영상

위의 방법과 마찬가지로 MATLAB에서 im2bw를 가지는데, 이는 아래와 같이 일반적 문장을 이용한 any data type의 영상을 문턱치 처리할 수 있다.

```
>> im2bw(r,0.43);
>> im2bw(b,0.39);
```

0 ~255 = 0 ~ 1

여기서 레벨은 0과 1 사이의 값(한계값 포함)이고, 분수로 되는 그레이 값은 흰색으로 치환한다. 이 명령은 그레이스케일 영상, 칼러 영상 및 데이터 형식이 unit8, unit16 또는 double형의 indexed 영상에서 수행한다. 예를 들면 문턱치 처리된 쌀과 박테리아 영상은 위의 명령을 이용하여 얻을 수 있다.

im2bw 함수는 해당 레벨이. 그레이 값이 그 영상의 형태를 가지도록 자동적으로 척도변환하고 다음에 첫 번째 방법으로 문턱치 처리를 수행한다.

배경에서 물체를 분리하는 것과 같이 문턱치 처리는 영상에 숨어 있는 모양을 보여주는 매우 간단한 방법이다. 예를 들면 영상 paper.tif는 모든 그레이 값들이 매우 높기 때문에 모두 희게 보인다. 그러나 높은 레벨에서 문턱치 처리를 하면 훨씬 의미가 있는 영상을 만든다.

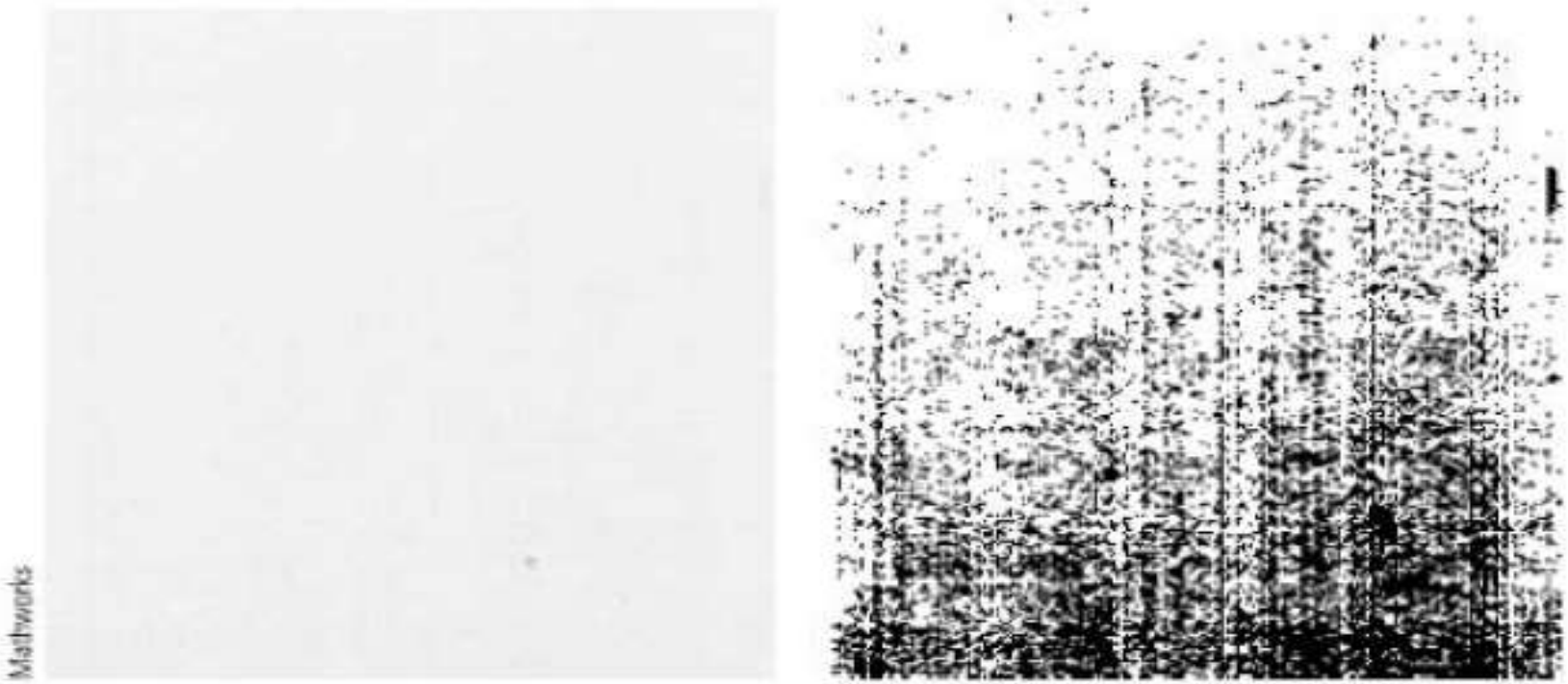


그림 9.3 종이 영상과 문턱치 처리 후 결과

이를 위해 아래의 명령을 이용하여 그림 9.3의 영상을 얻을 수 있다.

```
>> p=imread('paper1.tif');
>> imshow(p),figure,imshow(p>241)
```

9.2.2 더블 문턱치 처리

두 개의 값 T_1과 T_2를 선택하고 문턱치 처리를 아래와 같이 적용한다.

$$\text{a pixel becomes} \begin{cases} \text{white if its gray level is between } T_1 \text{ and } T_2, \\ \text{black if its gray level is otherwise.} \end{cases}$$

위의 방법으로 아래와 같이 단순한 변화를 구현할 수 있다.

`X>T1 & X<T2` T1 ~ T2 흰 백색

이 기호는 논리적 "and"이므로, 이 결과는 두 개의 부등식이 만족될 때에만 1이 된다. 다음 일련의 명령들은 spine.tif라는 indexed 영상의 8bit 그레이로 만드는 것으로 아래와 같이 시작 한다

인덱스, 컬러맵

```
>> [x,map]=imread('spine.tif');
>> s=uint8(    ind2gray(x,map)); 그레이 영상으로 변경
>> imshow(s),figure,imshow(s>115 & s<125) 115~125 사이면 백색
```

그 출력은 그림 9.4에 보였다. 더블 문턱치 처리는 단일 문턱치 처리로는 구별할 수 없는 척추(spine)의 미묘한 특징을 나타낼 수 있다. 아래와 같이 im2bw를 이용하여 유사한 결과를 얻을 수 있다.

```
>> imshow(im2bw(x,map,0.45)&    im2bw(x,map,0.5))
```

그러나 indexed 영상을 취급할 때 모든 여분의 계산을 포함하므로 계산이 매우 느리다

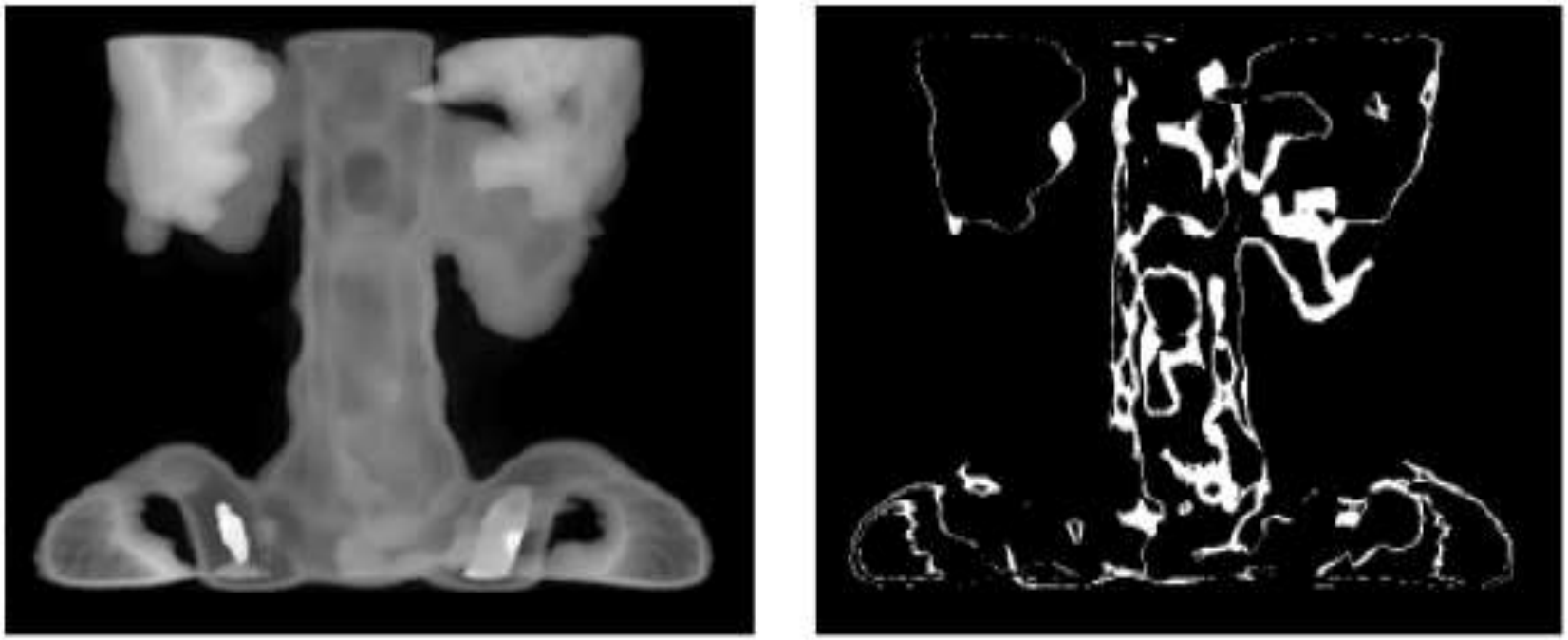


그림 9.4 척추(spine) 영상과 더블 문턱치 처리 영상

9.3 문턱치 처리의 응용

문턱치 처리를 다음과 같이 유용하게 적용할 수 있다.

1. 영상에서 반드시 필요한 부분만 남기고 불필요한 부분을 제거할 때 사용한다. 예를 들면 '쌀'과 '박테리아' 영상에서 모든 그레이 레벨 정보를 제거하고 쌀과 박테리아 부분만 2진 blob(영역)으로 축소하였다. 그러나 이 정보는 사이즈, 모양 및 blob의 수를 관찰할 수 있다.

2. 숨겨진 미세한 부분을 나타낼 때 사용한다. 이것은 종이나 척추 등의 영상에서 적용할 수 있고, 그레이 레벨이 포함된 유사성 때문에 미세한 성분이 불분명하다. 이러한 경우 특정한 부분을 나타낼 때 유용하다.

3. 교재나 그림으로부터 변화하는 배경을 제거하고자 할 때 적용 가능하다. text.tif라는 영상에서 변화하는 배경을 시험하기 위해 불규칙한 배경에서

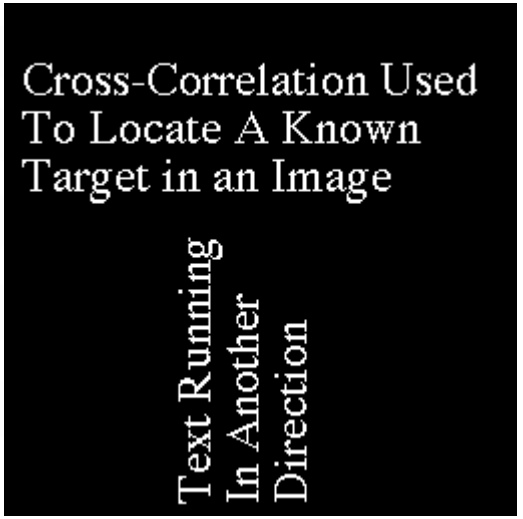
실행한다. 몇 가지 간단한 MATLAB 명령으로 아래와 같이 쉽게 구현할 수 있다.

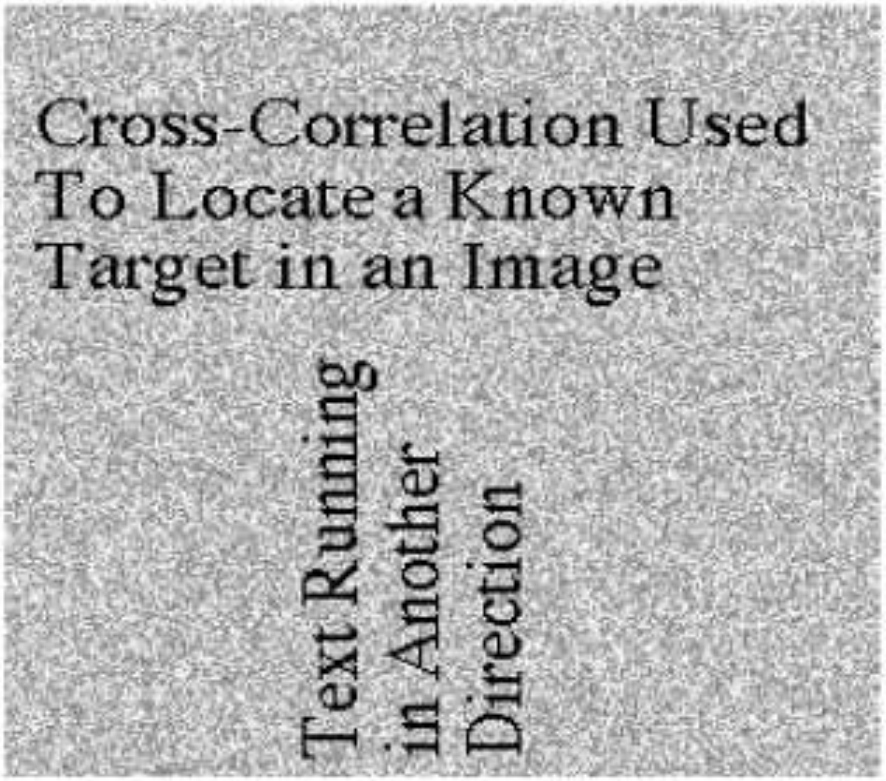
이의 첫째 명령은 rand 함수(0.0~1.0사이의 균일하게 불규칙한 수를 발생하는 매트릭스)를 사용하고, 이 불규칙한 수가 127~255가 되도록 척도변환 한다. 그 후에 어두운 배경에서 흰 text(글자)를 보여주는 영상을 읽는다.

3번째 명령은 여러 가지를 처리하는데, not(t)는 흰 배경에 있는 검은 text(글자)로 된 영상을 반전시킨다. 즉, double은 매트릭스가 연산 동작으로 사용할 수 있도록 숫자의 형을 변화시킨다. 최종적으로 그 결괴는 불규칙 매트릭스에 곱해지고, 그 전체가 디스플레이를 위해 unit8로 변환된다. 이 결과를 그림 9.5에 보였다. 이 영상을 문턱치 처리하고 아래와 같이 디스플레이 하면 그림 9.5의 우측과 같다.

```
>> imshow(tr>100)
```

```
r=rand(256)*128+127;
t=imread('text.tif');  imshow(t);
tr=uint8(r.*double(not(t)));
figure(2), imshow(tr);
figure(3), imshow(tr >100)
```





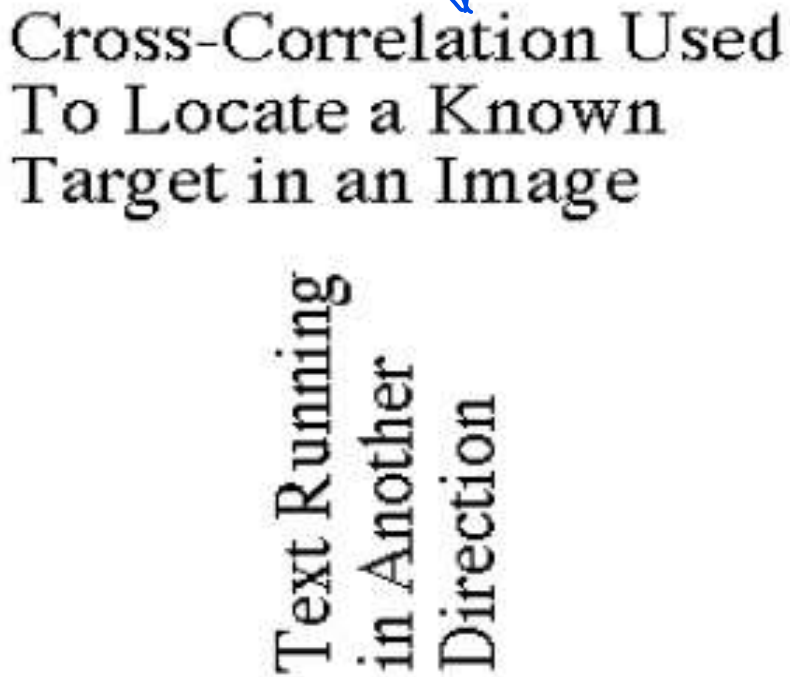


그림 9.5 변화하는 배경에서의 text 영상과 문턱치 처리 결과

9.4 적절한 문턱치의 선정

문턱치 처리를 이용하는 중요한 이유는 배경으로부터 물체를 분리하는 것이다. 그 후에 그 물체의 사이즈를 측정하거나 물체의 개수를 셀 수 있다. 분명히 이 처리과정의 성패는 적당한 문턱치 레벨의 결정에 좌우된다. 값을 너무 낮게 선정하면 물체의 사이즈가 감소되고 그 개수도 줄어든다. 반대로 너무 높게 선정하면 너무 많은 배경정보를 포함하게 된다.

nodules1.tif의 영상을 예를 들면, im2bw 함수를 이용하여 문턱치의 값을 0<t<1 범위에서 여러 가지를 아래와 같이 시도해 보자.

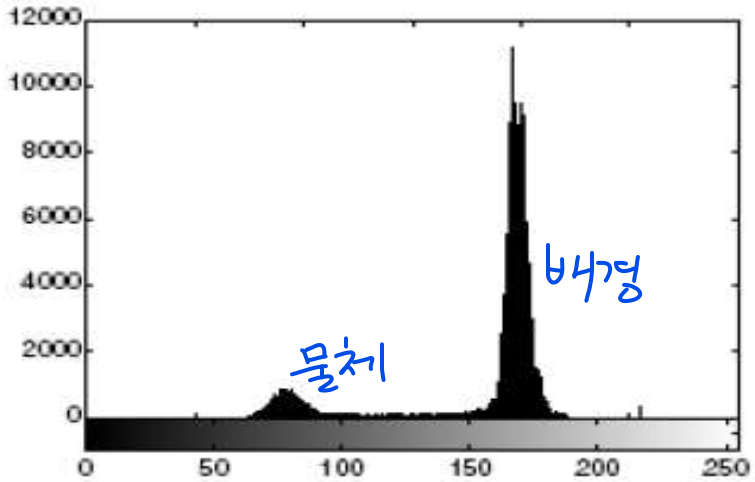
```
>> n=imread('nodules1.tif');
>> imshow(n);
>> n1=im2bw(n,0.35);
>> n2=im2bw(n,0.75);
>> figure,imshow(n1),figure,imshow(n2)
```

모든 영상은 그림 9.6에 보였다. 분명하게 구분할 수 있는 점이 존재한다면 히스토그램을 조사해보는 방법이 있다. 가끔 이 방법을 사용하지만, 반드시 할 필요는 없다.

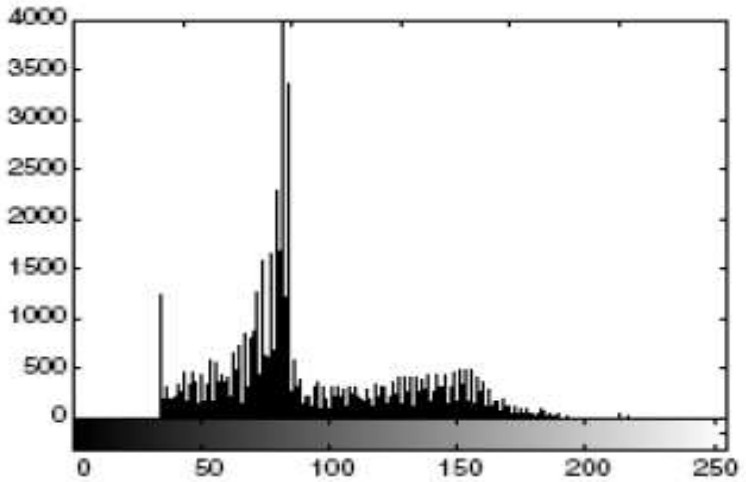
그림 9.7은 여러 가지 영상의 히스토그램을 보여준다. 각 경우에 영상은 배경위에 물체들로 구성되어 있다. coin과 nodule 영상에서 중간 정도의 위치에 히스토그램을 분리할 수 있지만, rice와 bacteria 영상은 그 선택이 분명치 않다.

n: Original image　　n1: Threshold too low　　n2: Threshold too high

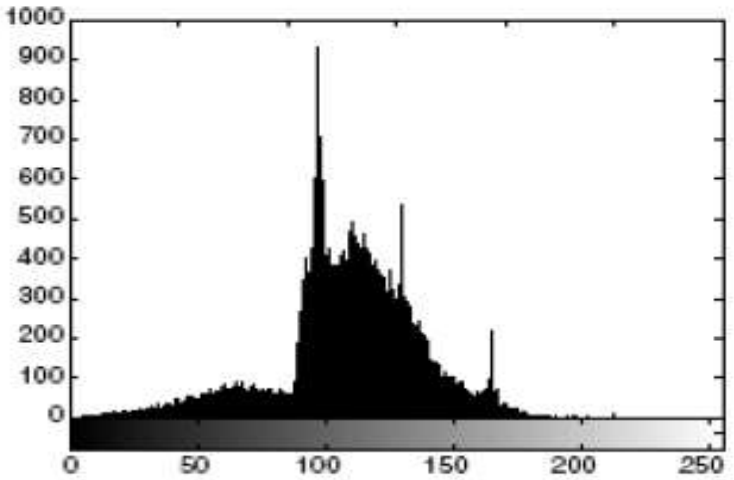
그림 9.6 nodule영상의 문턱치 처리 결과



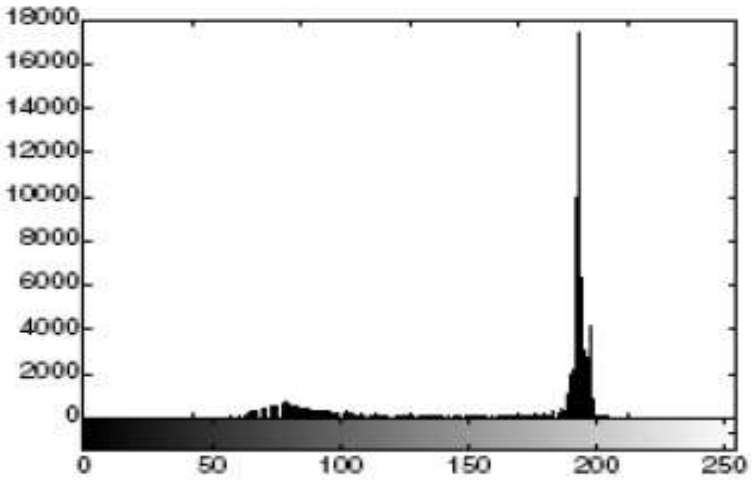
Nodules image : nodules.tif



Rice image : rice.tif



Bacteria image : bacteria.tif



Coins image : eight.tif

그림 9.7 여러 영상의 히스토그램

일반적인 문제는 물체와 배경의 히스토그램이 서로 중첩된다는 점이다. 이러한 경우 각 개별적인 히스토그램의 사전 지식이 없이는 분리할 수 있는 점을 찾기가 어렵다. 물체와 배경의 히스토그램이 각각 정규분포를 이룬다고 가정하면 그림 9.8과 같이 구별할 수 있다. 히스토그램이 서로 교차하는 위치에 문턱치를 선택한다.

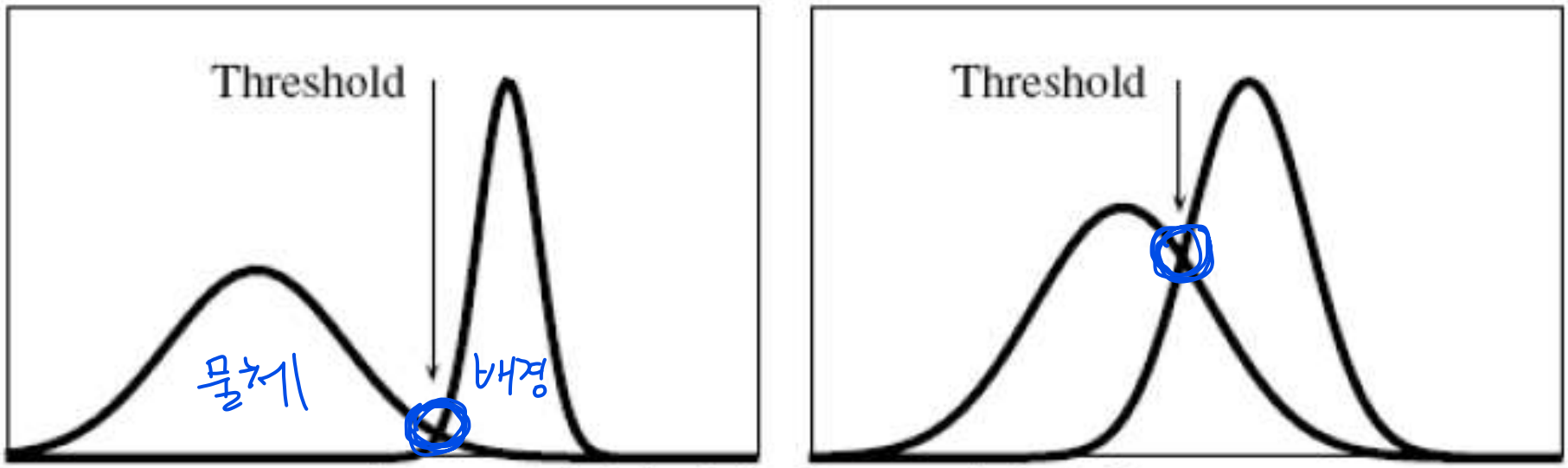


그림 9.8 문턱치에 의한 히스토그램의 분리

실제로, 히스토그램이 그림 9.8과 같이 분명히 정의되지 않더라도 최적의 문턱치를 선택하기 위한 몇 가지의 자동적인 방법이 필요하다. 그 중 하나가 확률분포로서 영상의 히스토그램을 아래와 같이 묘사하는 것이다.

$$p_i = n_i / N,$$

여기서 n_i는 그레이레벨 i를 가지는 화소의 수이고, N은 총 화소 수이며, p_i는 그레이레벨 i인 화소의 확률이다. 문턱치를 레벨 k로 할 때 아래와 같이 정의 할 수 있다.

왼쪽 면적

$$\omega(k) = \sum_{i=0}^{k} p_i$$

오른쪽 면적

$$\mu(k) = \sum_{i=k+1}^{L-1} p_i,$$

여기서 L은 그레이척도의 수이고, 따라서 L−1은 그레이 최대값이다. 정의에 의해 아래와 같이 표현할 수 있다.

$$\omega(k) + \mu(k) = \sum_{i=0}^{L-1} p_i = 1.$$

$\omega(k)$와 $\mu(k)$ 사이의 차분을 최대화하는 k를 구하기를 원한다.

최적 문턱치를 구하는 이 방법은 Otsu's method라고 하며, MATLAB 함수 graythresh로서 구현되었다. 4개의 영상 nodules, rice, bacteria 및 coins를 로서 이 방법을 아래와 같이 시도하였다. im2bw를 이용하여 이를 영상에 적용하였다. 이 결과를 그림 9.9에 보였고, 각 영상에 대한 결과가 만족스럽게 얻어졌다.

```
>> tn=graythresh(n)

tn =

    0.5804

>> r=imread('rice.tif');
>> tr=graythresh(r)

tr =

    0.4902

>> b=imread('bacteria.tif');
>> tb=graythresh(b)

tb =

    0.3765

>> e=imread('eight.tif');
>> te=graythresh(e)

te =

    0.6490
```

nodule 영상

```
>> imshow(im2bw(n,tn))
>> figure,imshow(im2bw(r,tr))
>> figure,imshow(im2bw(b,tb))
>> figure,imshow(im2bw(e,te))
```

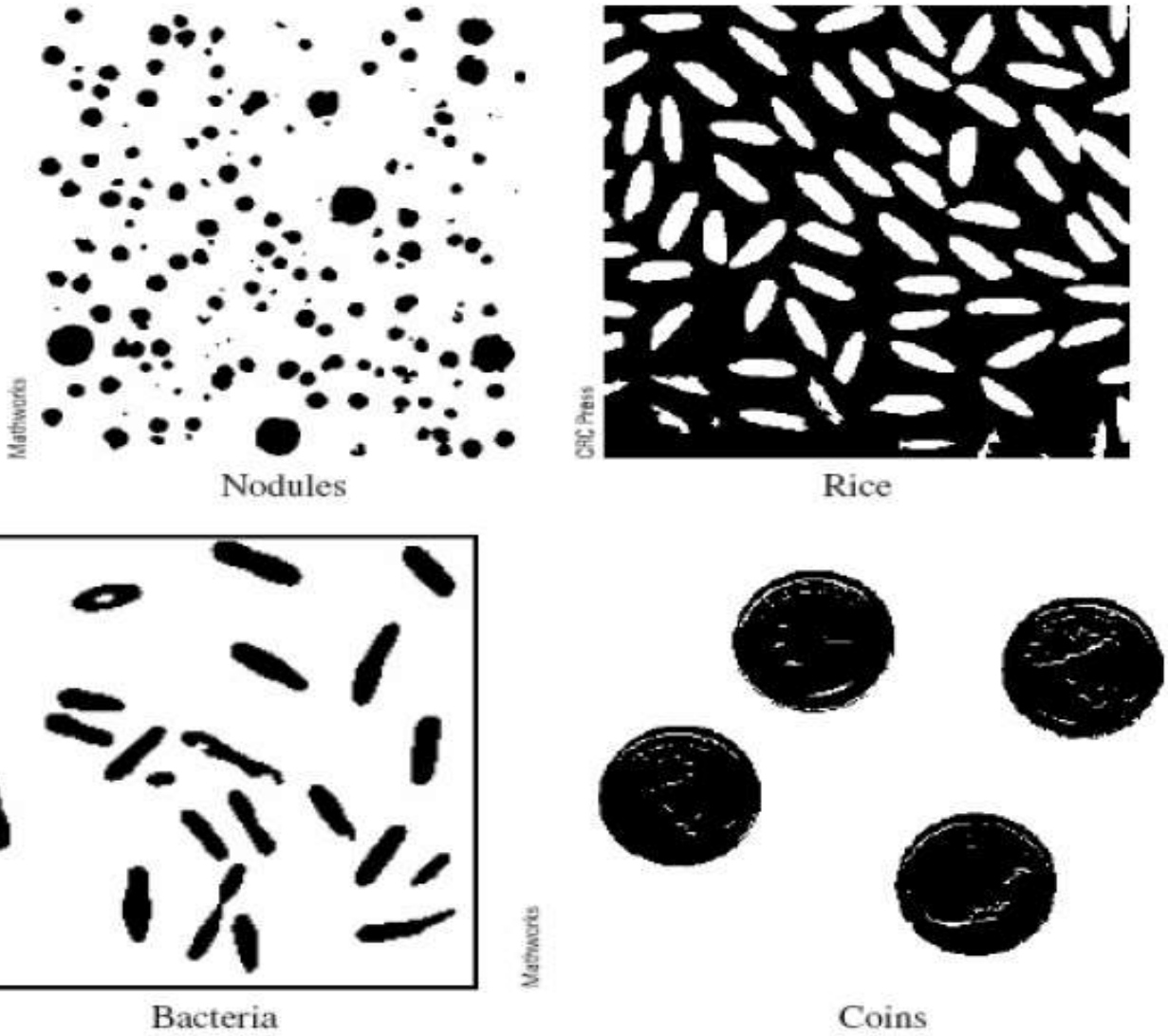


그림 9.9 graythresh 함수를 이용한 문턱치 처리 결과

9.5 적응 문턱치 처리

가끔 단일 문턱치로서 물체를 완전히 분리할 수 없는 경우가 있다. 이는 물체와 배경이 변하는 경우이다. 예를 들면 원(circle)영상을 입력하여, 원 부분과 배경 부분에 걸쳐 밝기가 변하는 환경으로 아래와 같이 만든다.

```
>> c=imread('circles.tif');
>> x=ones(256,1)*[1:256];
>> c2=double(c).*(x/2+50)+(1-double(c)).*x/2;
>> c3=uint8(255*mat2gray(c2));  % mat2gray : 행렬을 0~1 의 gray 영상으로 변환
```

그림 9.10에 graythresh를 아래와 같이 이용한 문턱치 처리를 보였다.

```
>> t=graythresh(c3)

t =

    0.4196

>> ct=im2bw(c3,t);
```

보는 바와 같이 결과는 특히 좋지 않은데, 물체의 전체가 배경에서 분리되지 않음을 알 수 있다. 다른 문턱치를 사용하더라도 그 결과는 비슷하다.

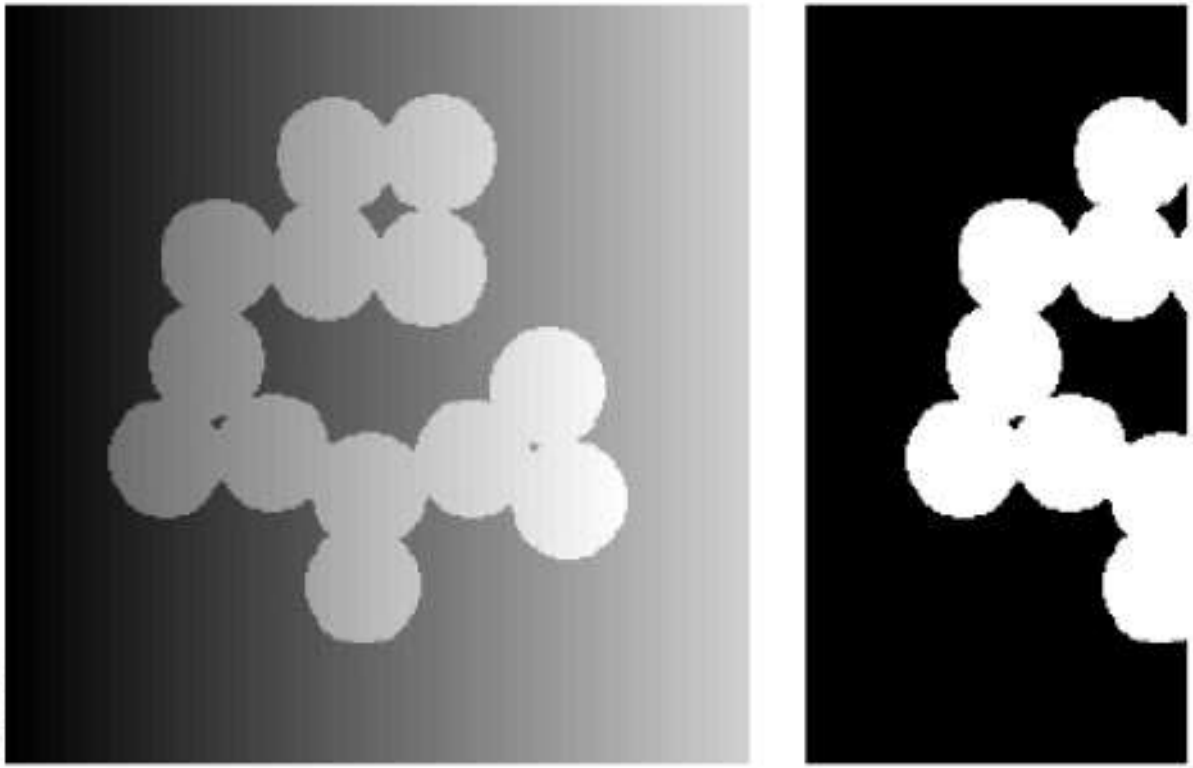
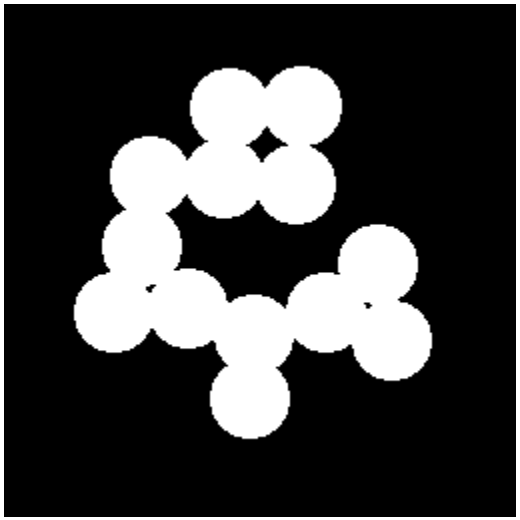


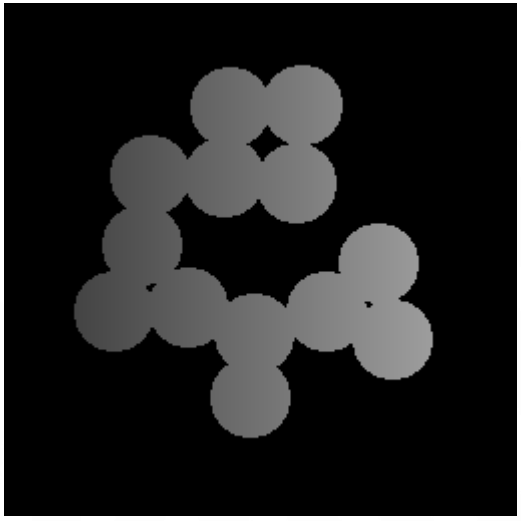
그림 9.10 circle 영상 (a)와 그 문턱치 처리 결과 (b)

Original



imshow(x/255)

kk=double(c).*(x/2+50); imshow(kk/255);



kk=(1-double(c)).*x/2; imshow(kk/255)

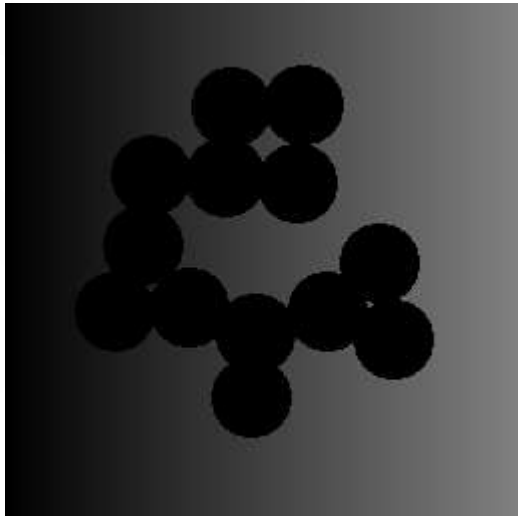
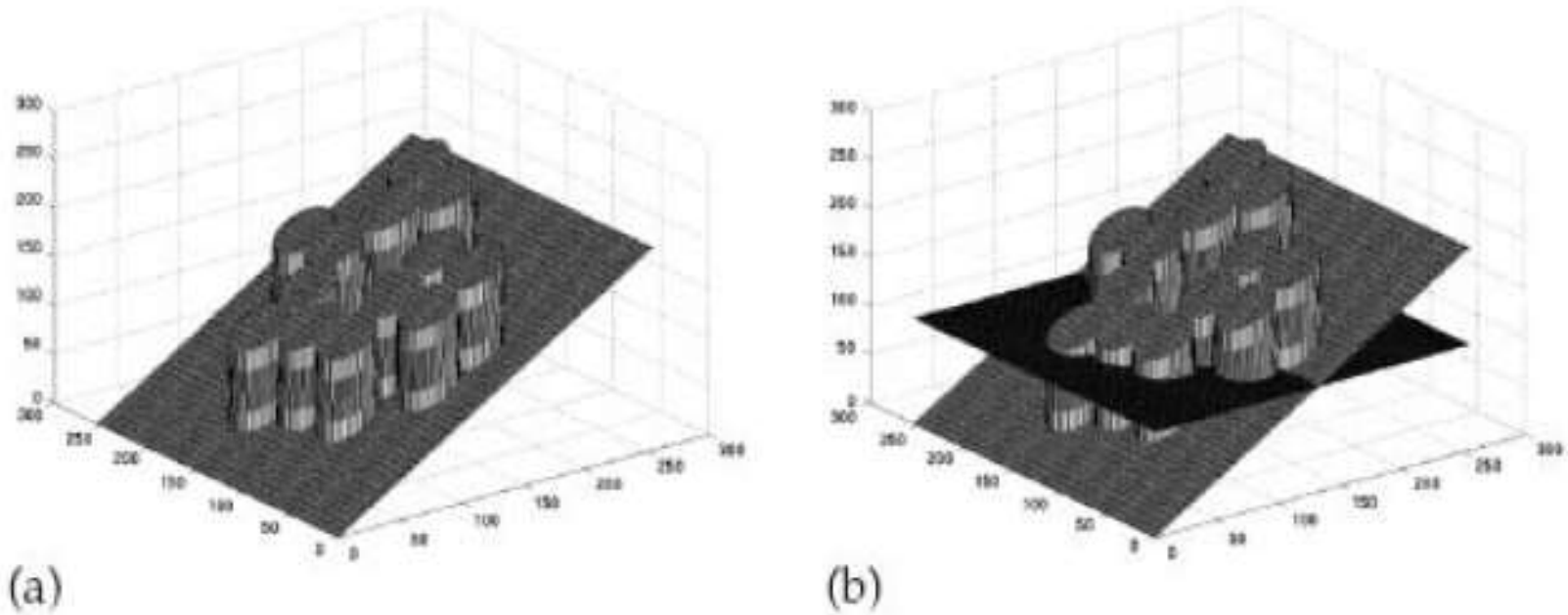


그림 9.11은 단일 문턱치로 분리되지 않는 이유를 나타내었다. 이 그림에서 영상은 함수로 나타나는데, 문턱치는 수평면으로 우측에 볼 수 있다. 평면의 어느 위치에서도 배경에서 원을 분리할 수 없게 된다.

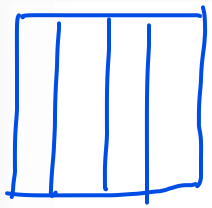


(a) (b)

그림 9.11 함수형태의 문턱치 처리 (a)함수적 영상 (b) 문턱치 처리 시도

행, 열

```
>> p1=c3(:,1:64);
>> p2=c3(:,65:128);
>> p3=c3(:,129:192);
>> p4=c3(:,193:256);
```



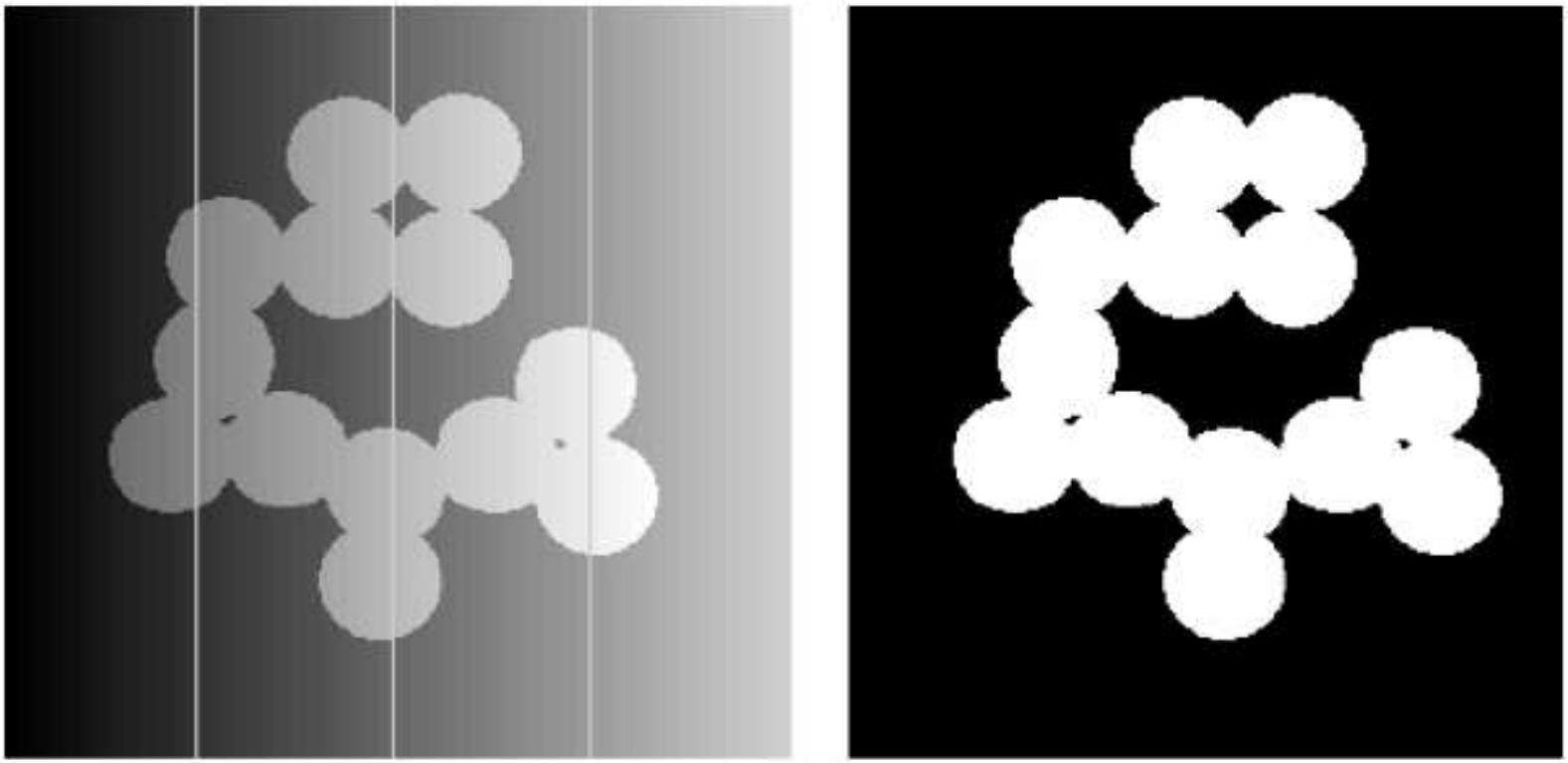
쪼개서 처리하는 작전
(적응 문턱치 처리)

그림 9.12는 영상을 조각낸 것을 보이고, 각 조각에 대하여 아래와 같이 문턱치 처리하였다

```
>> g1=im2bw(p1,graythresh(p1));
>> g2=im2bw(p2,graythresh(p2));
>> g3=im2bw(p3,graythresh(p3));
>> g4=im2bw(p4,graythresh(p4));
```

그 결과를 아래와 같이 단일 영상으로 디스플레이 하였다.

```
>> imshow([g1 g2 g3 g4])
```



(a) (b)

그림 9.12 적응 문턱치 처리 (a)4개의 조각영상 (b)각 분리 처리한 문턱치 처리 결과

이 결과는 그림 9.12 (b)와 같다.

9.6 에지 검출(Edge detection) 물체를 구분하는 또 다른 방법

에지는 영상에서 가장 유용한 정보를 포함한다. 에지는 영상에서 물체의 사이즈를 측정하기도 하고, 배경에서 특정한 물체를 분리하기도 하며, 물체를 인식 또는 분류하는데 적용된다. 에지를 구하는 알고리듬이 많이 존재한다. 에지를 구하기 위한 일반적인 MATLAB 명령은 아래와 같다.

input 마스크 임계값 처리(얼마나 차이나면 Edge로 인정하는가)

```
edge(image, 'method', parameters . . . )
```
% method: mask, parameter: 문턱치

여기서 parameter는 그 방법에 따라 다르게 사용된다. 이 절에서 기본 필터링 방법을 이용하여 에지영상을 만드는 법을 보이고, MATLAB의 에지 함수를 논의한다.

에지는 주어진 문턱치를 초과하는 화소값들로 국부적 불연속으로 느슨하게 정의되는데, 화소값들의 차분을 관측할 수 있다. 예를 들면 그림 9.13을 생각하자.

51	52	53	59
54	52	53	62
50	52	53	68
55	52	53	55

(a)

50	53	155	160
51	53	160	170
52	53	167	190
51	53	162	155

(b)

두 데이터를 비교하는 것 : 미분

급격하게 변하는 걸 찾는 것

그림 9.13 화소들의 블록 : 에지 검출

그림에서 (b)의 경우 2번째 열과 3번째 열의 화소값의 차분이 뚜렷이 나타남을 알 수 있다. 이와 같이 이웃하는 화소의 차분이 생길 때 인간의 눈은 그레이값의 차이를 구분하게 된다. 이 차분을 에지로 검출하게 된다.

9.7 미분과 에지

9.7.1 기본적 정의

그림 9.14의 영상을 생각하자. 영상의 왼쪽에서 시작하여 오른쪽으로 횡단하면서 그레이 값을 그린다고 가정하자. 두 개의 에지가 나타나는데, 그레이 값들이 서서히 변하는 ramp에지와 급하게 변하는 step에지가 있다.

그림 9.14를 구성하는 함수를 f(x)라 하고, 그 도함수를 f'(x)를 그린다. 그림 9.15는 이를 보여준다. 기대한 바와 같이 화소의 값들이 일정한 부분은 모두 0이 되고, 화소의 차분이 있는 부분에서는 0이 되지 않는다.

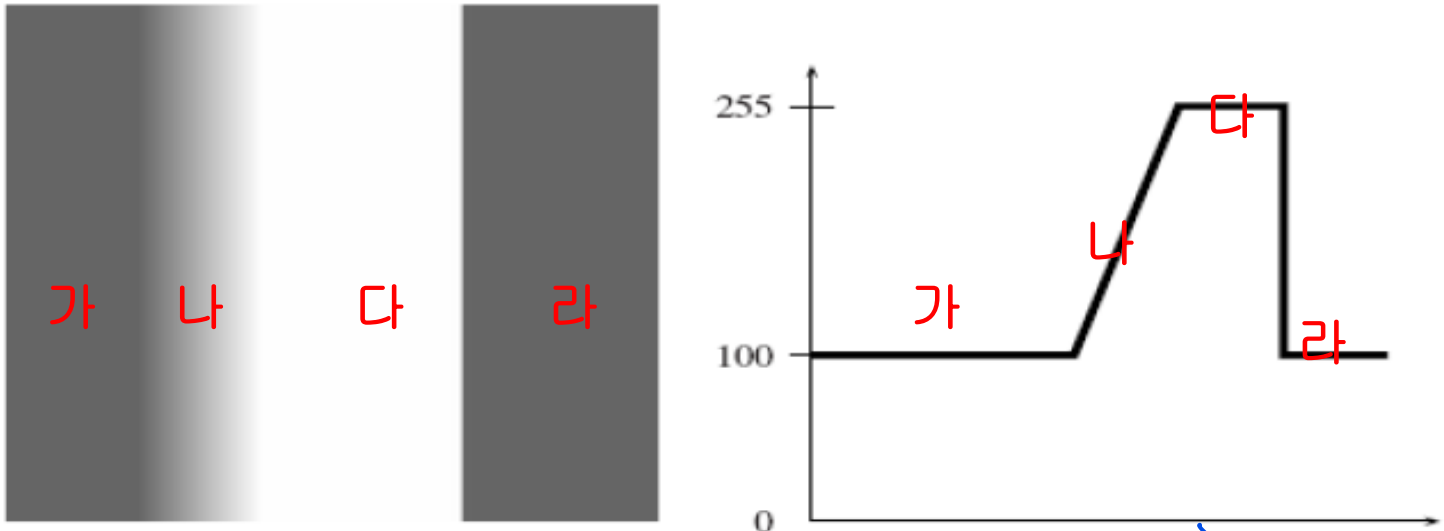


그림 9.14 에지와 그 구조

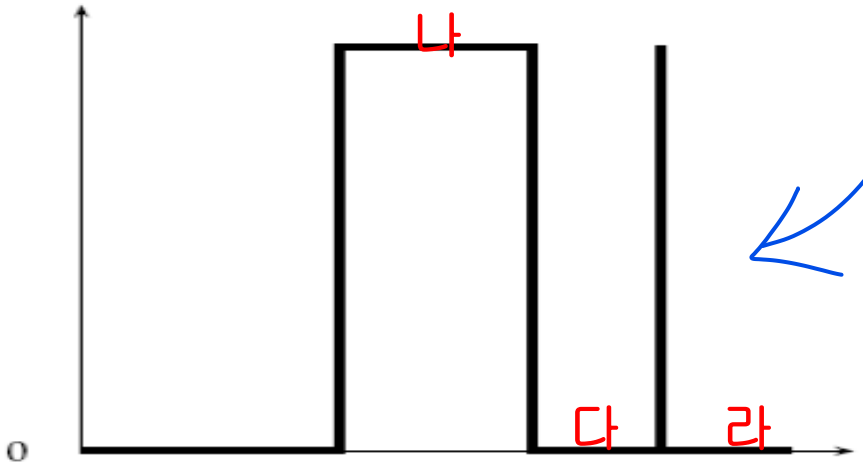


그림 9.15 에지 구조의 미분 결과

변화율을 그래프로 표현

대부분의 에지탐색 동작은 미분에 기초를 두는데, 이산 영상에 대해 연속적 도함수를 적용한다. 먼저 아래와 같이 도함수를 정의한다.

$$\frac{df}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}.$$

영상에서, h의 최소 양의 값이 1이므로 2개의 이웃하는 화소의 지수(index)사이에 차분이 f(x+1)−f(x)로 존재하므로 도함수의 표현에 이산 형태는 아래와 같이 표현할 수 있다.

$$\lim_{h \to 0} \frac{f(x) - f(x-h)}{h}, \quad \lim_{h \to 0} \frac{f(x+h) - f(x-h)}{2h}$$

이의 이산 형은 아래와 같이 쓸 수 있다.

$$f(x) - f(x-1), \quad \frac{(f(x+1) - f(x-1))}{2}.$$

2차원 영상에 대하여 편미분을 사용할 수 있다. 이의 표현은 gradient(기울기) 이고, 아래의 표현과 같이 정의되는 벡터이다.

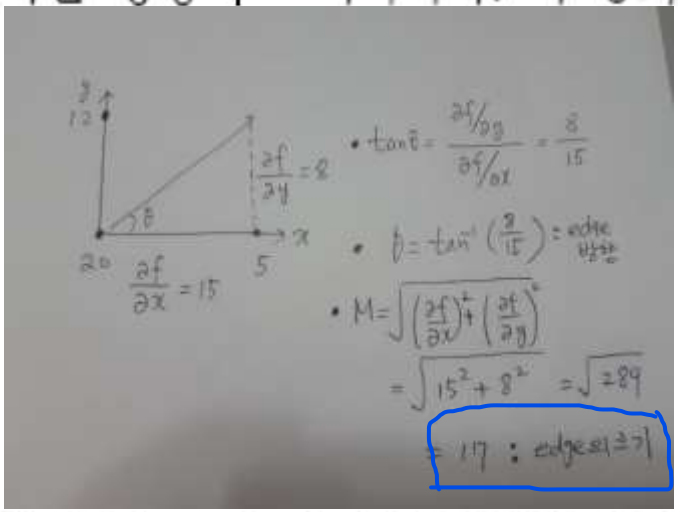
$$\left[\frac{\partial f}{\partial x} \quad \frac{\partial f}{\partial y}\right]',$$

이는 함수 f(x,y)의 점이 가장 크게 증가하는 방향의 벡터이다. 이 증가 방향은 아래의 각도로 나타내고,

$$\tan^{-1}\left(\frac{\partial f/\partial y}{\partial f/\partial x}\right),$$

그 크기는 아래와 같이 나타낸다.

$$\sqrt{\left(\frac{\partial f}{\partial x}\right)^2+\left(\frac{\partial f}{\partial y}\right)^2}.$$



대부분의 에지검출 방법은 기울기의 크기를 구하고 이 결과에 문턱치 처리를 적용한다.

9.7.2 몇 가지의 에지검출 필터

차분을 이용하여 척도(scaling)를 그대로 유지하는 수평과 수직방향의 필터를 내면 아래와 같다.

$$[-1 \quad 0 \quad 1] \qquad \text{and} \qquad \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}.$$

가로으로 Edge 검출하는 필터

세로 "

$$P_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}.$$

이 필터와 상반되는 것은 수평에지를 구하는 것으로 아래와 같다.

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

위 필터를 에지검출을 위한 prewitt 필터라 한다.

p_x 및 p_y가 영상에 적용되어 그레이 값을 얻었다면 그 기울기의 크기는 아래와 같이 얻을 수 있다.

$\sqrt{p_x^2 + p_y^2}$. Edge 의 크기

그러나 실제로는 아래의 2가지 중 하나를 구하는 것이 적절하다.

$\max\{|p_x|, |p_y|\}$

$|p_x| + |p_y|$.

예를 들면 그림 9.16의 IC회로의 영상을 택하여 MATLAB 아래와 같이 읽어서 처리해 보자.

```
>> ic=imread('ic.tif');
```

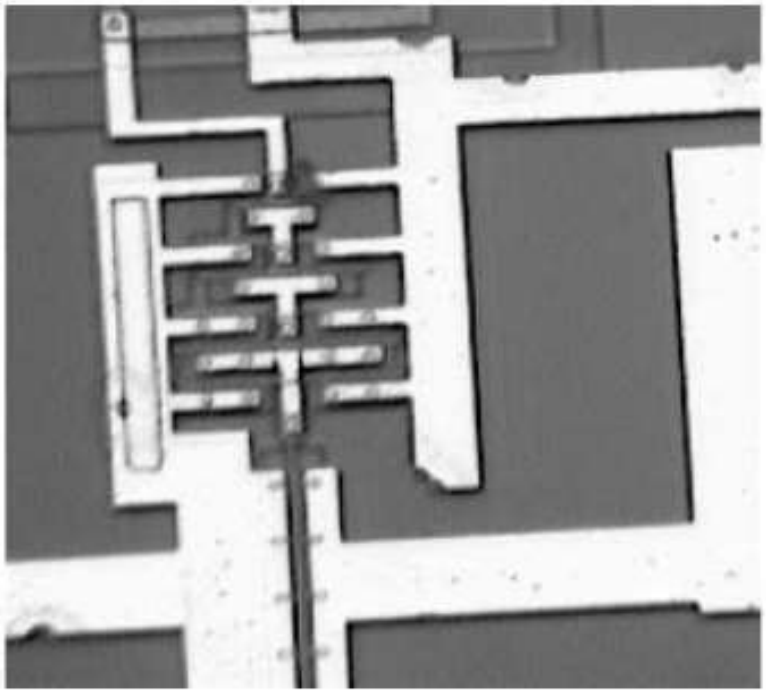


그림 9.16 IC 회로 영상

그림 9.17에 각각 p_x 및 p_y를 구한 것을 보였다. 그림 (a)는 아래의 명령으로 만들어졌다.

```
>> px=[-1 0 1;-1 0 1;-1 0 1];
>> icx=filter2(px,ic);
>> figure,imshow(icx/255)
```

좌우대칭 필터(수직)

ic는 input

또 그림 (b)는 아래의 명령으로 구한 것이다.

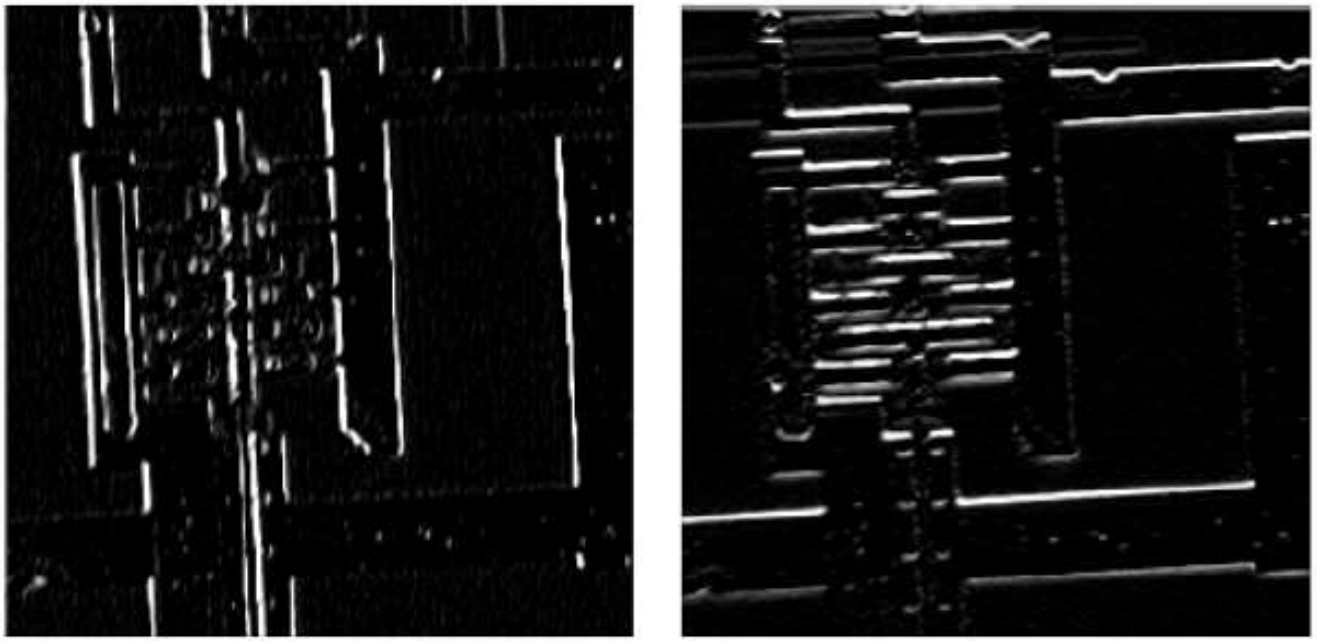


그림 9.17 prewitt필터 적용 결과 (a) 수직 필터 (b) 수평 필터

여기서 필터 p_x는 수직에지의 최고치를, p_y는 수평에지의 최고치를 나타낸다. 아래와 같이 모든 에지를 포함하는 그림을 만들 수 있다.

전치행렬, 행과 열을 바꿔주는 것이다.

```
>> py=px';
>> icy=filter2(py,ic);
>> figure,imshow(icy/255)
```

이 결과는 그림 9.18과 같다. 이것은 원래 그레이스케일 영상인데, 문턱치 처리에 의해 에지를 포함하는 이진영상으로 만들어 진다. 그림 9.18 (b)는 아래의 명령을 적용한 결과이다.

```
>> edge_p=sqrt(icx.^2+icy.^2);
>> figure,imshow(edge_p/255)
```

```
>> edge_t=im2bw(edge_p/255,0.3);
```

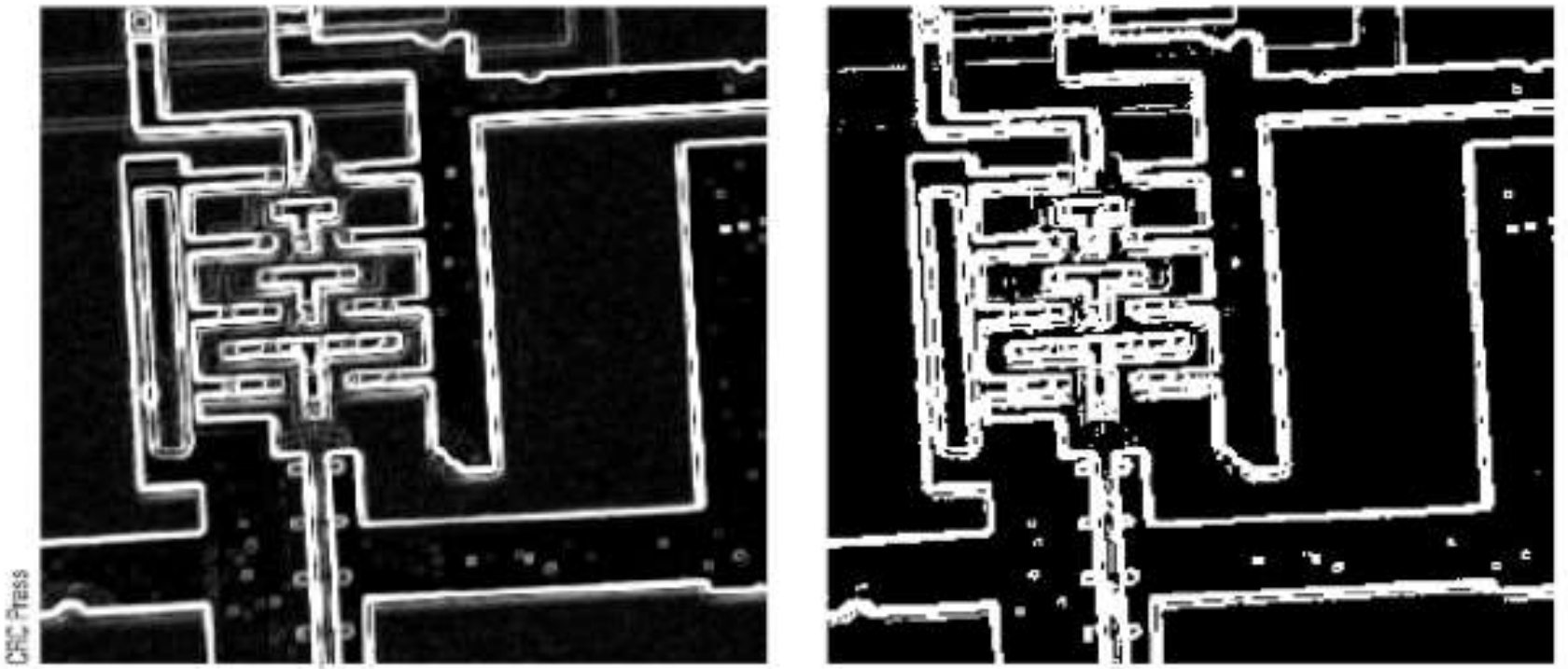


그림 9.18 IC 회로의 모든에지검출 결과(수직성분+수평성분)

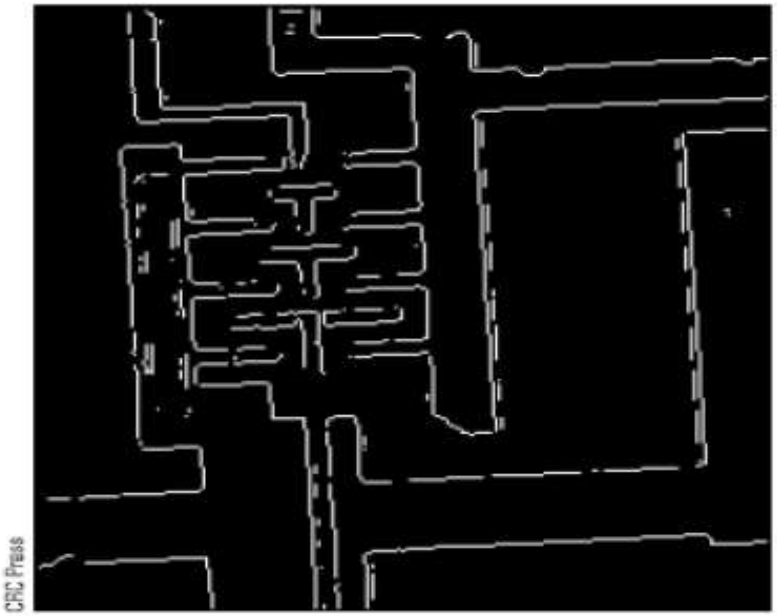
Imshow(edge_p/255);

Imshow (im2bw (edge_p/255, 0.3));

아래의 명령을 사용하면 직접 prewitt 필터로 에지를 구할 수 있다.

```
>> edge_p=edge(ic,'prewitt');
```

이 에지함수는 양방향 필터링을 취하고 적당한 문턱치 레벨을 선택한다. 이 결과는 그림 9.19와 같다. 그림 9.18 (b)와 그림 9.19는 서로 다르다.



0.95가 default

그림 9.19 prewit필터의 에지검출의 option적용 결과

이것은 여분의 처리를 실행하고, 필터에 대한 제곱의 합을 다시 평방근을 취하여 얻은 결과이다.

이와는 조금 다른 에지검출 필터로서 아래와 같이Robert 대각-기울기 필터가 있다.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

그리고, Sobel 필터는 아래와 같다.

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} -1 & -2 & 1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}.$$

```
>> edge_r=edge(ic,'roberts');
>> figure,imshow(edge_r)
```

```
>> edge_s=edge(ic,'sobel');
>> figure,imshow(edge_s)
```

이들 각 영상은 문턱치를 변경하여 변화시킬 수 있다.

이들 필터에서 Sobel 필터가 가장 좋은 특성을 가지고, 잡음이 존재하는 영상에서도 비교적 좋은 에지를 얻을 수 있다.

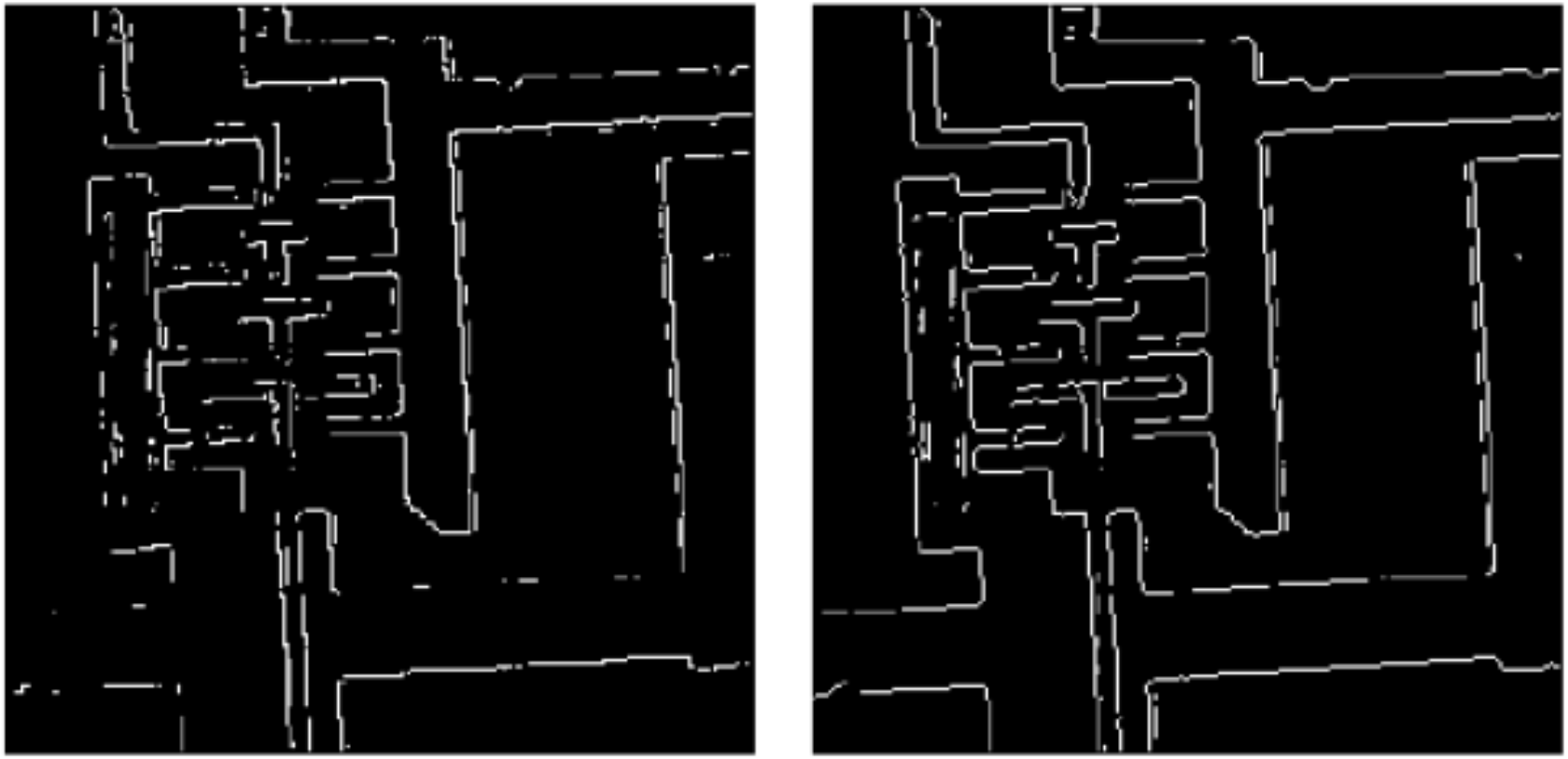


그림 9.20 Robert 및 Sobel 필터의 에지검출 결과
(a) Robert 필터 (b) Sobel 필터

9.8 2차 도함수

수평 + 수직 = 1차 도함수 필터
수평과 수직을 동시에 = 2차 도함수 필터

9.8.1 Laplacian

또 다른 하나의 에지검출 방법은 2차 도함수를 적용하는 것이다. 양방향의 2차 도함수의 합을 Laplacian이라 하고 아래와 같이 표현한다.

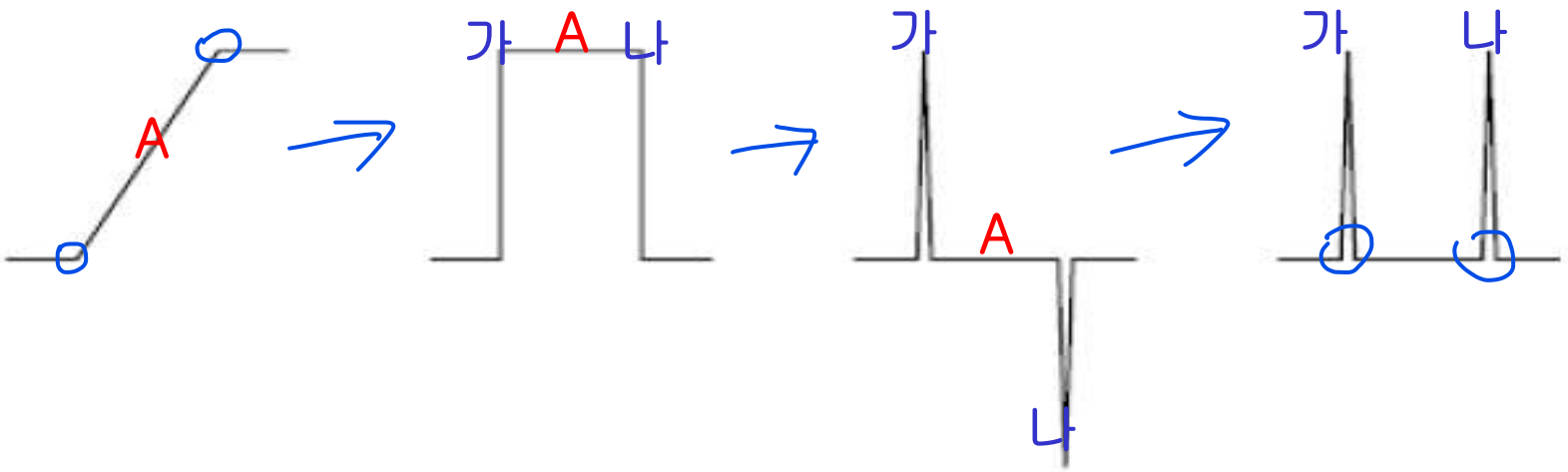
$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

이를 필터로 구현하면 아래와 같다.

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

이것은 이산 라플라시안으로 알려져 있다. 라플라시안은 1차 도함수 방법보다 등방향성필터(isotropic filter)의 특성을 가지므로 장점을 가진다. 이는 회전 불변을 의미한다. 즉, 라플라시안을 영상에 적용한 후에 회전을 시켜도 영상을 먼저 회전시킨 다음 라플라시안을 적용한 것과 그 결과는 같다. 이것은 에지검출의 이상적인 필터를 구성한다. 그러나 주된 문제점은 모든 2차 도함수의 필터는 잡음에 매우 민감하다는 것이 단점이다.

2차 도함수가 에지에 어떤 영향을 주는가를 알기 위해서 그림 9.5의 화소 값들의 도함수를 구하면 그 결과는 그림 9.21과 같게 된다.



The edge First derivative Second derivative Absolute values

그림 9.21 에지함수의 2차 도함수

라플라시안(절대치 또는 제곱을 취한 후)는 2중의 에지를 구한다. 예를 들면 아래와 같이 명령을 MATLAB에 입력한다고 하자.
이 결과는 그림 9.22와 같다.

```
>> l=fspecial('laplacian',0);
>> ic_l=filter2(l,ic);
>> figure,imshow(mat2gray(ic_l))
```

라플라시안은 노이즈에
민감하므로, '영 교차' 필요

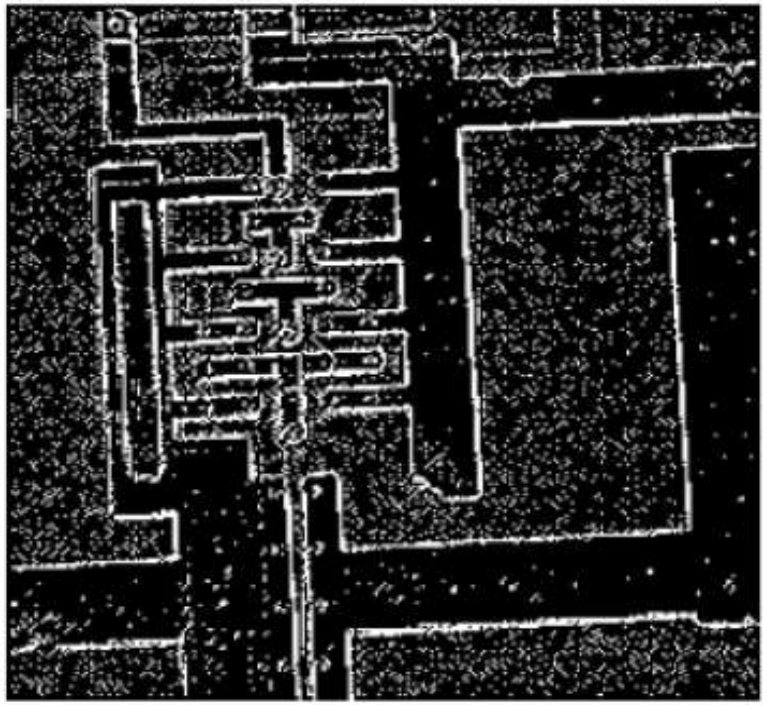


그림 9.22 이산 라플라시안 필터링 처리 결과

9.8.2 영 교차(Zero Crossing)

라플라시안을 위한 더욱 적당한 방법은 영 교차의 위치로서 에지의 위치를 구하는 것이다. 그림 9.21에서 에지의 위치는 필터의 값이 0이 되는 곳에 주어진다. 일반적으로 이 점은 필터의 결과의 부호가 변하는 곳이다. 예를 들면 그림 9.23 (a)의 단순한 영상에 대하여 라플라시안 마스크의 필터링 결과를 그림 9.23 (b)에 보였다.

다음 중 하나를 만족하는 화소가 되는 필터링 영상에서 영 교차를 정의한다.

1. 음의 그레이 값을 가지고, 이웃하는 화소의 그레이 값이 직교하는 양의값이 되는 경우
2. 음과 양의 값을 가지는 화소 사이에 0의 값을 가지는 경우

50	50	50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50	50	50
50	50	200	200	200	200	200	200	50	50
50	50	200	200	200	200	200	200	50	50
50	50	200	200	200	200	200	200	50	50
50	50	200	200	200	200	200	200	50	50
50	50	50	50	200	200	200	200	50	50
50	50	50	50	200	200	200	200	50	50
50	50	50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50	50	50

(a)

-100	-50	-50	-50	-50	-50	-50	-50	-50	-100
-50	0	150	150	150	150	150	150	0	-50
-50	150	-300	-150	-150	-150	-150	-300	150	-50
-50	150	-150	0	0	0	0	-150	150	-50
-50	150	-150	0	0	0	0	-150	150	-50
-50	150	-300	-150	0	0	0	-150	150	-50
-50	0	150	300	-150	0	0	-150	150	-50
-50	0	0	150	-300	-150	-150	-300	150	-50
-50	0	0	0	150	150	150	150	0	-50
-100	-50	-50	-50	-50	-50	-50	-50	-50	-100

(b)

그림 9.23 영상에서 영 교차의 위치 (a) 단순 영상 (b) 필터링 결과

그림 9.23 (b)에서 영 교차점이 회색처리 되어 있다. 여기서 부가적인 에지검출 방법을 소개한다. 즉, 라플라스 필터링 후에 영 교차를 처리한다. 이것은 에지의 영 교차 옵션으로 MATLAB에서 구현되는데, 주어진 필터 처리 후에 영 교차를 아래와 같이 처리한다.

```
>> l=fspecial('laplace',0);
>> icz=edge(ic,'zerocross',l);
>> imshow(icz)
```

이 결과를 그림 9.25에 보였다. 사실 이것은 매우 좋은 결과는 아니다. 왜냐하면 이 방법으로 만들어지는 에지는 너무 많은 그레이레벨 변화가 생기기 때문이다. 그들을 제거하기 위하여 먼저 가우시안 필터로 영상을 스므딩한다. 이는 에지검출에 대하여 Marr-Hildreth 방법이라는 다음의 단계에 따라 처리한다.

1. 가우시안필터로 영상을 스므딩처리를 한다.
2. 그 결과를 라플라시안필터로 회선(convolution)처리 한다.
3. 영 교차를 구한다.

이 방법은 가급적 생체학적 시각에 가깝게 하기 위한 에지검출법으로 고안되었다. 첫 두개의 단계는 LoG(Laplacian of Gaussian)필터를 만들기 위해 하나로 결합될 수 있다. 이 필터는 fspecial 함수로 만들 수 있다. zerocross의 에지 옵션이라는 여분의 파라메터가 없는 경우 필터는 아래와 같이 LoG필터가 얻어진다.

```
>> fspecial('log',13,2)
```

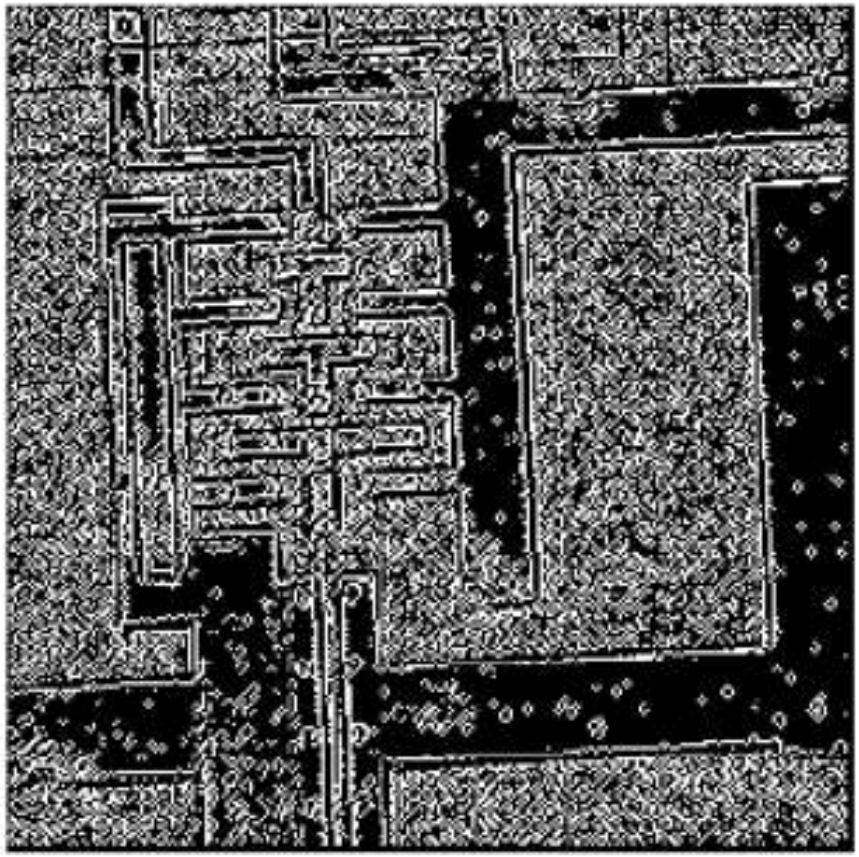
이것은 아래와 같은 명령을 의미한다.

```
>> edge(ic,'log');
```

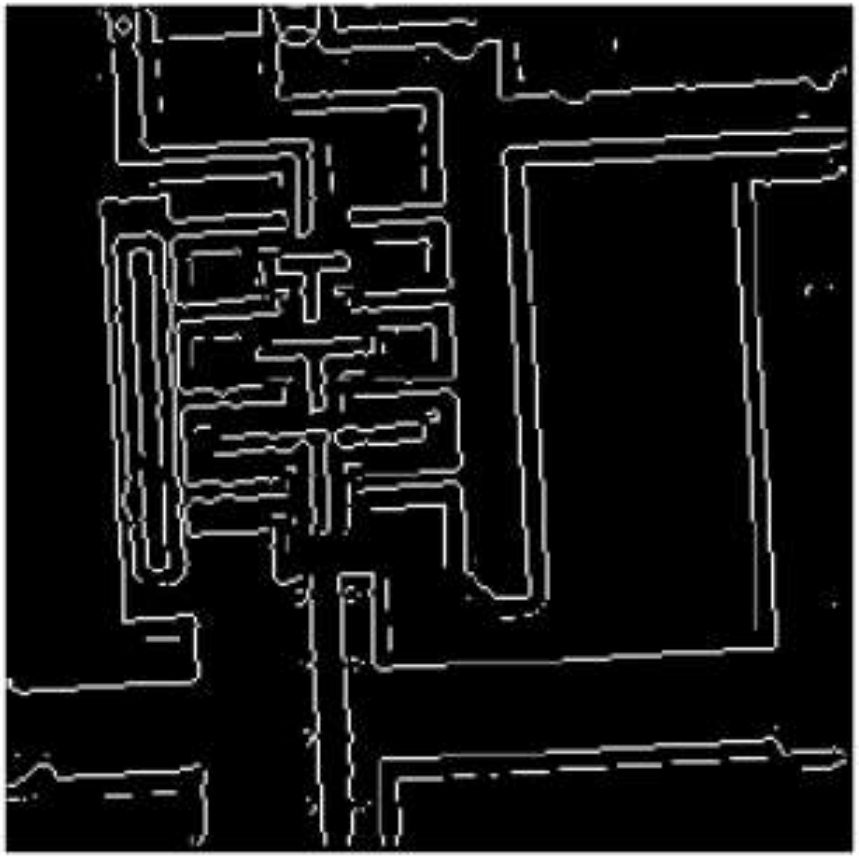
위 2개의 명령은 아래의 명령과 동일한 결과를 준다.

```
>> log=fspecial('log',13,2);
>> edge(ic,'zerocross',log);
```

사실 LoG와 zerocross 옵션은 동일한 에지 탐색 방법을 구현한다. LoG필터의 적용 후에 영 교차를 구한 결과는 그림 9.25(b)와 같다.



(a)



(b)

그림 9.25 영 교차를 이용한 에지검출 (a) 영 교차 (b) LoG 필터의 적용

9.9 Canny 에지검출기

1986년에 Jhon Canny라는 사람이 Canny edge detector를 구현한 또 다른 에지함수가 있다. 이것은 다음의 3가지 기준으로 고안되었다.

1. 낮은 에러율의 검출. 에지가 아닌 것을 제외한 모든 에지를 구한다.
2. 에지의 위치. 영상에서 실제 에지들 사이의 거리와 이 알고리즘으로 구해진 에지들을 최소화 한다.
3. 단일 응답. 이 알고리즘은 단지 하나의 에지가 존재할 때 다중 에지화소를 받아들여서는 안 된다.

Canny는 그의 알고리즘의 시작에 사용할 최상의 필터가 가우시안(스므딩을 위해)이고, 그 뒤에 1차원 가우시안 도함수의 적용을 보였다.

이 필터는 에지에 대하여 잡음의 스므딩과 가능한 에지들의 후보를 구하는 효과를 가진다. 이 필터를 분리할 수 있기 때문에 먼저 수직형(column) 필터로 수직방향으로 적용하고, 다음으로 수평형(row) 필터 로 수평방향으로 적용할 수 있다. 이렇게 하여 에지 영상을 구성하기 위해 2개의 결과를 합성할 수 있다. 이는 앞에서 적용한 2차원 필터를 적용하는 것보다 계산에서 훨씬 효과적이다.

따라서 여기서는 다음과 같은 처리한다.

1. 영상을 읽는다.
2. Gaussian Filter 를 적용한다(Noise 제거, Smoothing)
3. Roberts, Sobel, Prewitt 등을 적용하여 수평, 수직 Edge 를 구한다.
4. 아래 식을 이용하여 edge의 값을 구한다.

$$xe = \sqrt{x1^2 + x2^2}.$$

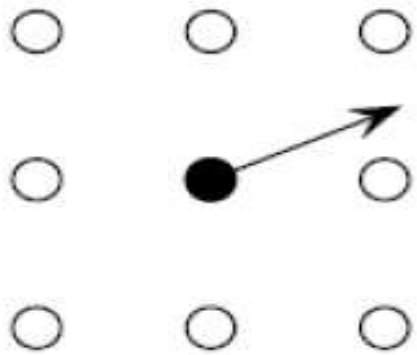
그리고나서 임계값처리

이렇게 한다고 해서 하나의 필터로 표준 에지검출을 하는 것보다 훨씬 많은 것을 실행한다고는 볼 수 없다.

다음 단계는 non-maximum suppression(비 최대치 억제)처리 이다. 에지화소들만 유지시키기 위해 에지영상 xe를 문턱치 처리하고 그 외는 제거한다. 그러나 문턱치 처리만으로 만족할 만한 결과를 만들 수 없다. 각각의 화소 p는 에지화소로 고려해야 할 방향성 Φ_p(에지의 경사)를 가지며, p는 방향성 Φ_p에서 그 이웃 화소보다 더 큰 크기를 가져야 한다.

위 크기 xe를 계산하는 것처럼 아래와 같이 역 tan함수를 이용하여 에지 경사를 계산할 수 있다.

$$xg = \tan^{-1}\left(\frac{x2}{x1}\right).$$

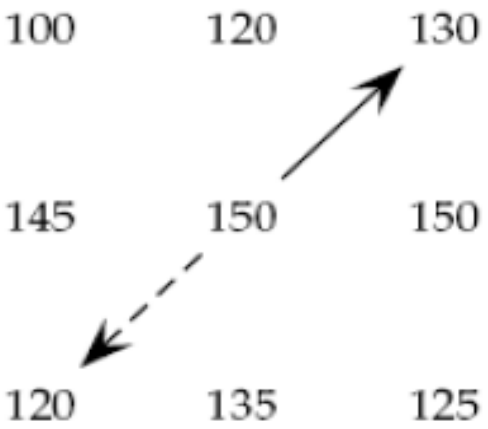


The edge direction at a pixel

그림 9.26 Canny 에지검출기에서 비 최대치 억제

2번째 접근은 값들 중의 하나를 0°, 45°, 90°, 및 135°로 양자화하고, 원래의 경사를 이 양자화한 점들에 대한 화소의 경사와 비교한다. 위치 (x,y)에서 경사를 $\phi(x,y)$라 하고, $\phi'(x,y)$를 얻기 위해 주어진 4개의 각도 중 하나가 되도록 $\phi(x,y)$를 양자화한다. (x,y)로부터 $\phi'(x,y)$와 $\phi'(x,y)+180°$ 방향으로 2개의 화소를 생각하자. 이들 중 에지의 크기가 현재 화소의 크기보다 더 크다면, 현재 화소를 삭제하기 위해 마크한다. 이러한 처리를 전체 영상에 걸쳐 처리한 후에 마크한 모든 화들을 삭제한다.

그림 9.28과 같이, 경사가 45°로 양자화된 1개 화소에 대하여 1개의 이웃 화가 있다고 가정하자. 점선 화살표는 현재 경사의 반대방향을 가리킨다. 이 그림에서 화살표의 끝에서 양측의 크기는 중심 크기보다 더 작다. 그래서 하나의 에지로서 중심 화소를 유지한다. 트러나 그들 중 하나가 150 이상이면 중심 화소를 삭제하기 위한 마크를 붙인다.



가운데가 제일 크면 Edge

그림 9.28 비 최대치 억제를 위한 양자화

비 최대치 억제처리 후에, 이진 에지영상을 얻기 위해 문턱치 처리를 할 수 있다. Canny는 단일 문턱치 처리 대신에 낮은 값 t_L 및 높은 값 t_H로 2개의 값을 사용하는 hysteresis thresholding(가설 문턱치 처리)라고 하는 기술을 제안하였다. t_H보다 더 큰 값을 가진 화소는 에지로 간주한다. 또한 에지에 인접하고 있는 $t_L \le p \le t_H$에 잇는 값도 에지화소가 될 수 있다.

Canny 에지검출기는 edge 함수의 canny 옵션으로 구현된다. 문턱치들을 입력하거나 아래와 같이 자동으로 선택할 수 있다.

```
>> [icc,t]=edge(ic,'canny');
>> t

t =

    0.0500    0.1250

>> imshow(icc)
```

이 결과를 그림 9.29에 보였다. 각각 다른 문턱치를 적용한 결과는 그림 9.30에 나타내었다. 상위의 문턱치를 보다 높게 할수록 더욱 적은 에지가 검출된다. 원래의 가우시안 필터의 표준편차를 변화시킬 수 있다.

Canny 에지검출기는 지금까지 설명한 에지검출들 중 가장 복잡하다. 그러나 에지검출기들 중에서 각광받는 것은 Parker에 의해 주어진 것인데, 고급 에지검출기로 알려져 있다.

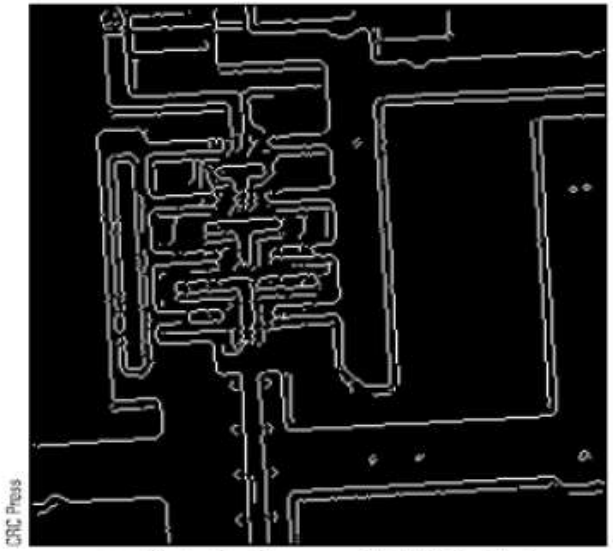
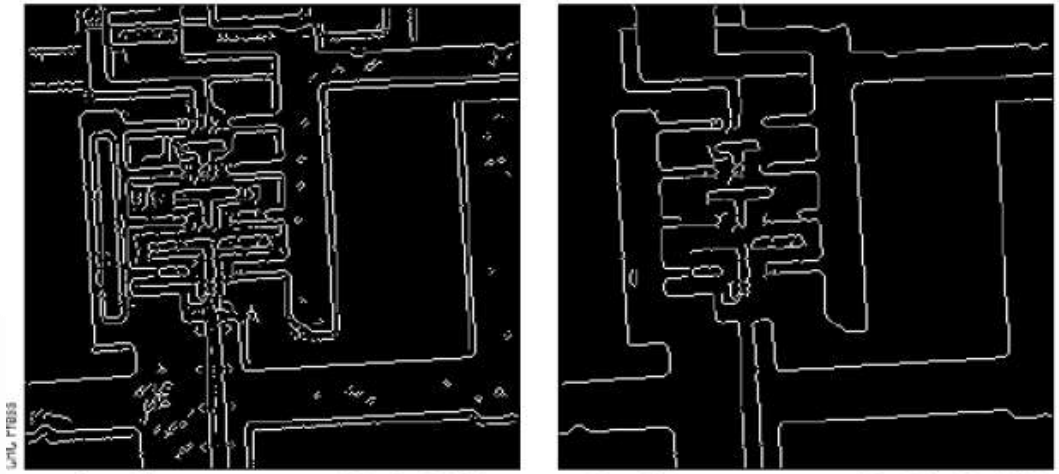


그림 9.29 Canny 에지검출 결과



edge(ic,'canny',[0,0.05])　　edge(ic,'canny',[0.01,0.5])

그림 9.30 각각 다른 문턱치를 적용한 Canny 에지검출 결과