MATLAB을 이용한

디지털 영상처리의 기초

10.1 서 론

수학적 형태학, 또는 형태학은 영상에서 형태를 해석하기 위한 영상처리의 특정 부분에 해당한다. 이진 영상의 조사에 대한 형태학적 기본 TOOL을 개발하고 이들 TOOL을 그레이 영상으로 확장하는 방법을 보인다. MATLAB에는 영상처리 TOOL-BOX에서 이진 형태학에 대한 많은 TOOL이 있고, 그레이스케일 형태학에 대해서도 사용할 수 있다.

10.2 기본 개념

수학적 형태학의 이론은 여러 가지 방법으로 발전될 수 있다. 점들의 집합으로 연산되는 한가지의 표준 방법을 채택한다. Haralick과 Shapiro에 의해 견고하고 상세한 내용을 다룬다.

제 10장 영상의 형태적 처리

10.2.1 이동(Translation)

A가 이진 영상에서 화소들의 집합이고, w=(x,y)가 특정 좌표의 점이라고 가정하자. 그러면 A_w는 방향 (x,y)만큼 이동된 집합 A이다. 즉, 아래와 같이 표현할 수 있다.

$$A_w = \{(a,b) + (x,y) : (a,b) \in A\}.$$

만큼 이동

예를 들면 그림 10.1에서 A는 교차형의 집합이고, w=(2,2)이다. 집합 A는 w에 주어진 값만큼 x와 y방향으로 이동된 것이다. 직각좌표계가 아니라 매트릭스를 이용한다. 그러므로 원점은 왼쪽 위에 있고, x는 세로방향, y는 가로방향으로 나타낸다.

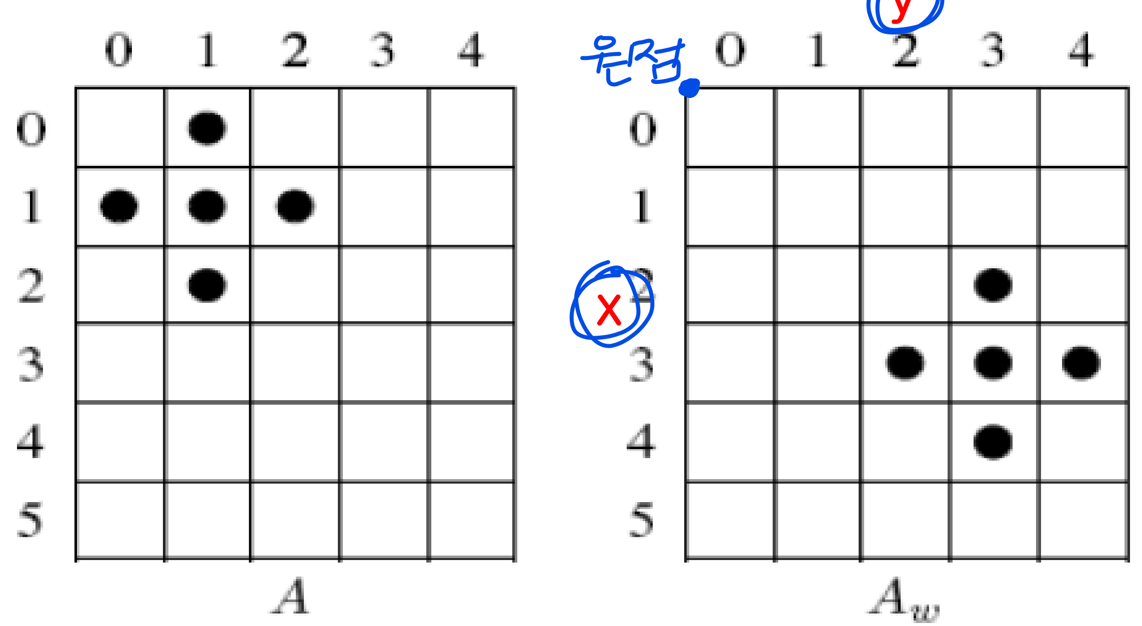


그림 10.1 이동(translation)

10.2.2 반사(reflection)

A가 화소들의 집합이면, 그 반사는 Â로 표시하고 원점에 대하여 A를 반사시켜 아래와 같이 얻는다.

$$\hat{A} = \{(-x, -y) : (x, y) \in A\}.$$

예를 들면 그림 10.2에서 흰 원과 검은 원이 서로의 반사를 나타내는 집합이다.

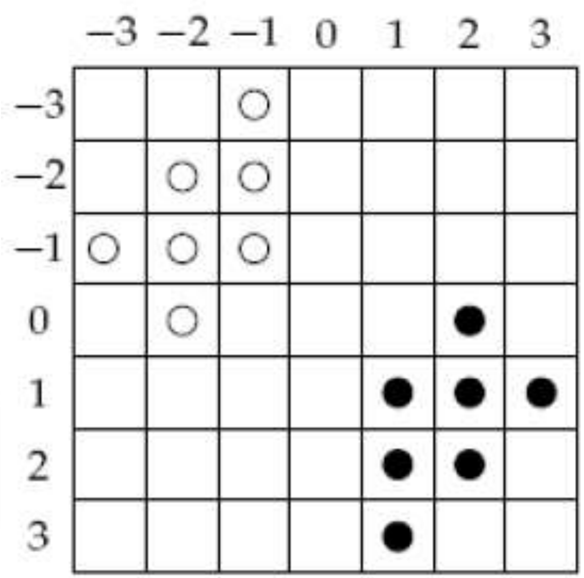


그림 10.2 반사(reflection)

10.3 팽창(Dilation)과 침식(Erosion)

팽창과 침식은, 여러 가지 연산들이 이들 2가지의 결합으로 이루어지는 형태학의 기본 연산이다.

10.3.1 팽창

구멍을 메꾸는 데 용이

A와 B는 화소들의 집합이고, B에 의한 A의 팽창은 A⊕B로 표시하고 아래와 같이 정의한다.

마스크

$$A \oplus B = \bigcup_{x \in B} A_x.$$

이것은 x의 모든 점은 B의 원소이고, 그 좌표만큼 A를 이동시키며 그 후에 이들 모든 이동결과를 합한다는 의미이다. 등가적 표현은 아래와 같다.

$$A \oplus B = \{(x,y)+(u,v) : (x,y) \in A, (u,v) \in B\}.$$

이 정의로부터 팽창은 아래와 같이 교환법칙이 성립된다.

$$A \oplus B = B \oplus A.$$

팽창의 예는 그림 10.3에 보였다. 이동의 다이아그램에서, 그레이 값의 사각형은 물체의 원점을 나타낸다. 물론, $A_{(0,0)}$ 는 A의 자신이다. 이 예에서 B의 좌표는 아래와 같고, A를 b의 각각의 좌표만큼 이동시킨다.

$$B = \{(0,0),(1,1),(-1,1),(1,-1),(-1,-1)\},$$ BP

일반적으로 A⊕B는 A에 있는 모든 점 (x,y)를 B에 대한 사본으로 치환하고, (x,y)에서 B의 (0,0)으로 넣어서 구해질 수 있다. 다시 말하면, A의 사본을 B의 모든 점 (u,v)에 치환할 수 있다.

팽창은 또한 Minkowski addition으로 알려져 있고, 더욱 자세한 내용은 전문서적을 참조 바란다.

그림 10.3과 같이 팽창은 물체의 크기를 증가시키는 효과를 가진다. 그러나 원래의 물체 A가 그 팽창인 A⊕B의 내부에 반드시 놓여 져야 할 필요는 없다. B의 좌표에 따라서 A⊕B는 A에서 멀리 떨어지는 경우도 있다. 그림 10.4는 이를 보여주는 예이다. 그림에서와 같이 B는 같은 모양을 가지지만, 위치가 다르다. 이 그림에서 B의 위치는 아래와 같다.

$B = \{(7,3),(6,2),(6,4),(8,2),(8,4)\}$ 10p

그러므로 팽창의 결과는 아래와 같다.

$$A \oplus B = A_{(7,3)} \cup A_{(6,2)} \cup A_{(6,4)} \cup A_{(8,2)} \cup A_{(8,4)}.$$

팽창에 대하여 일반적으로 A는 처리될 영상이고, B는 화소들의 작은 집합으로 간주한다. 이 경우에 B는 구조적 요소 혹은 커널(kernel)에 해당한다.

MATLAB에서 팽창은 다음 명령으로 수행할 수 있다.

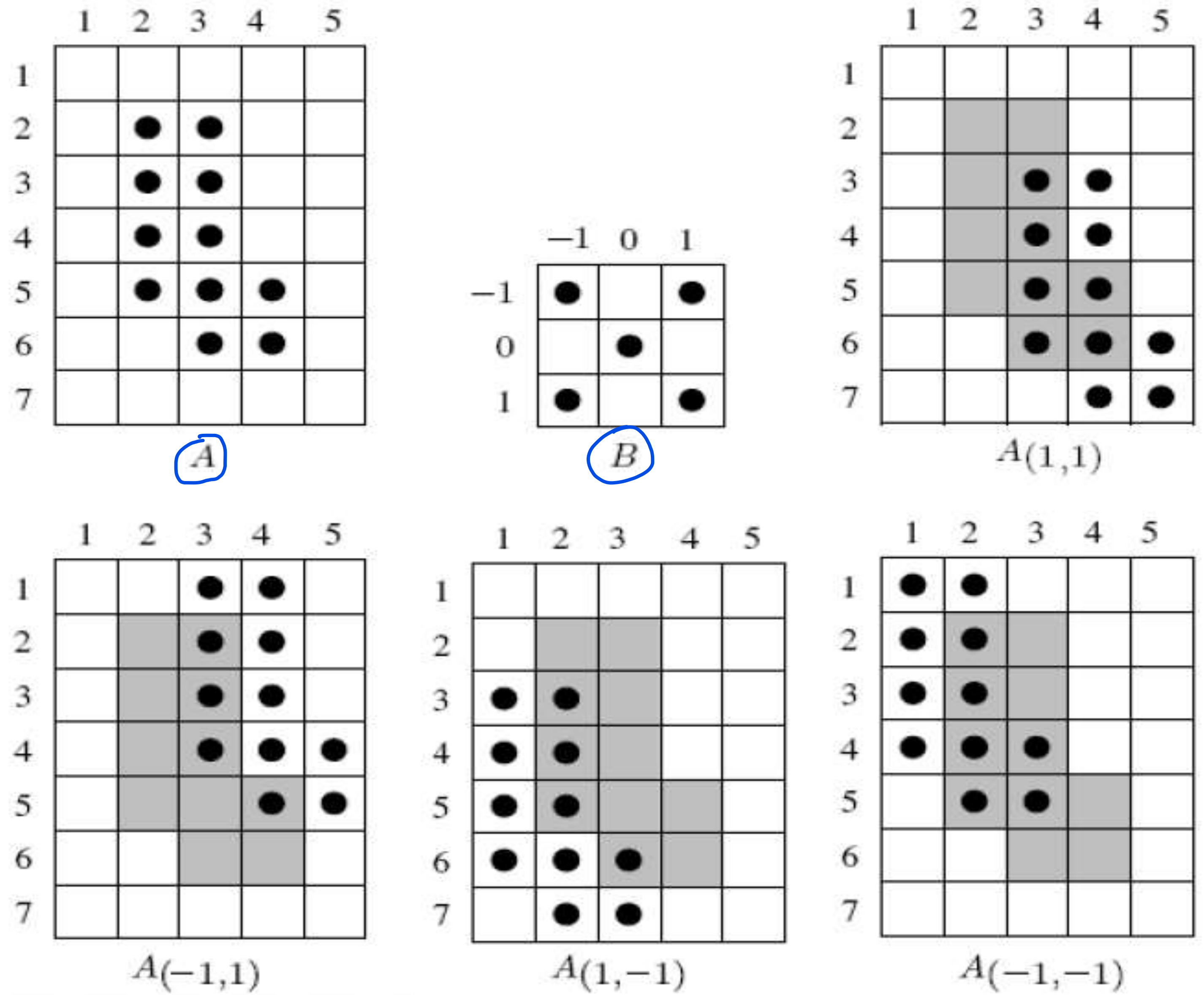
input mask

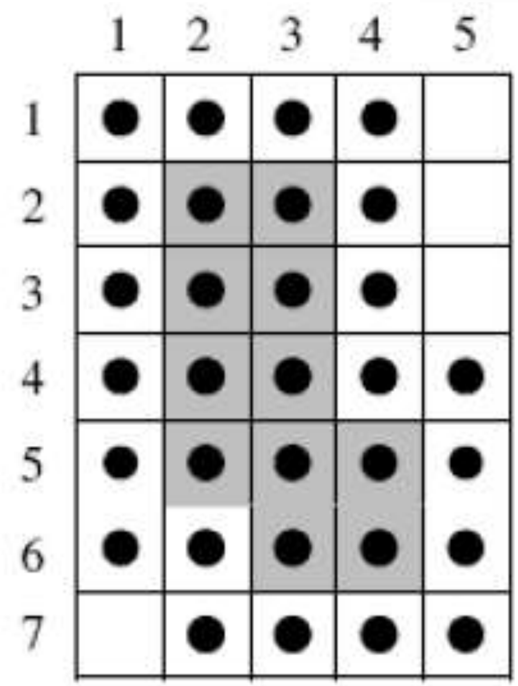
```
>> imdilate(image,kernel)
```

팽창에 대한 예를 보기 위해 아래의 명령을 실행한다.

```
>> t=imread('text.tif'); input
>> sq=ones(3,3); mask
>> td=imdilate(t,sq);
>> subplot(1,2,1),imshow(t)
>> subplot(1,2,2),imshow(td)
```

이 결과는 그림 10.5 (b)와 같다. 영상이 두터워진 결과를 알 수 있다. 이 두터워지는 현상은 팽창의 의미를 보여주는 것이다.


A
B
A(1,1)
A(−1,1)
A(1,−1)
A(−1,−1)



$A \oplus B$

결과

그림 10.3 팽창의 연산 과정

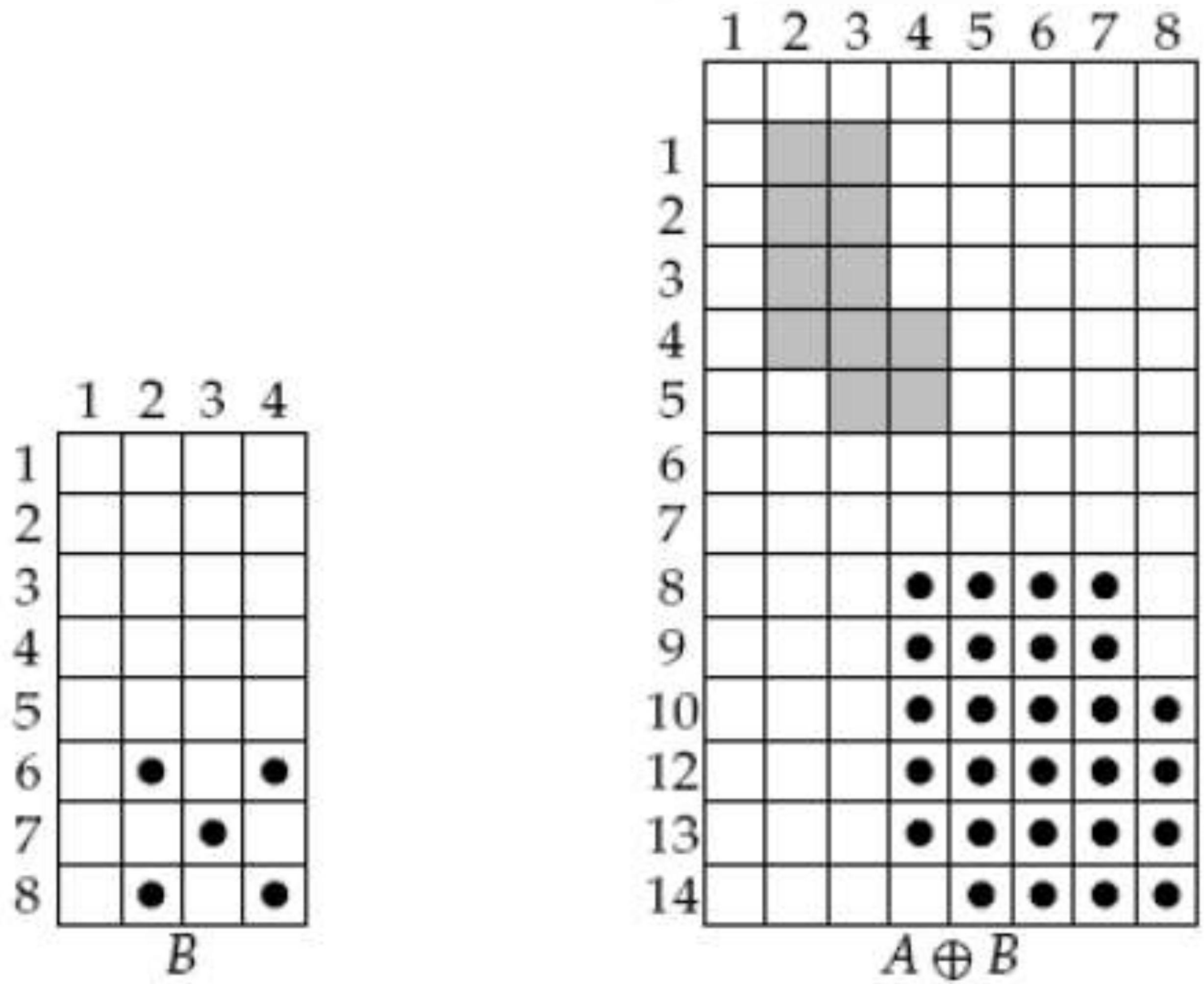


그림 10.4 $A \not\subseteq A \oplus B$인 팽창의 예

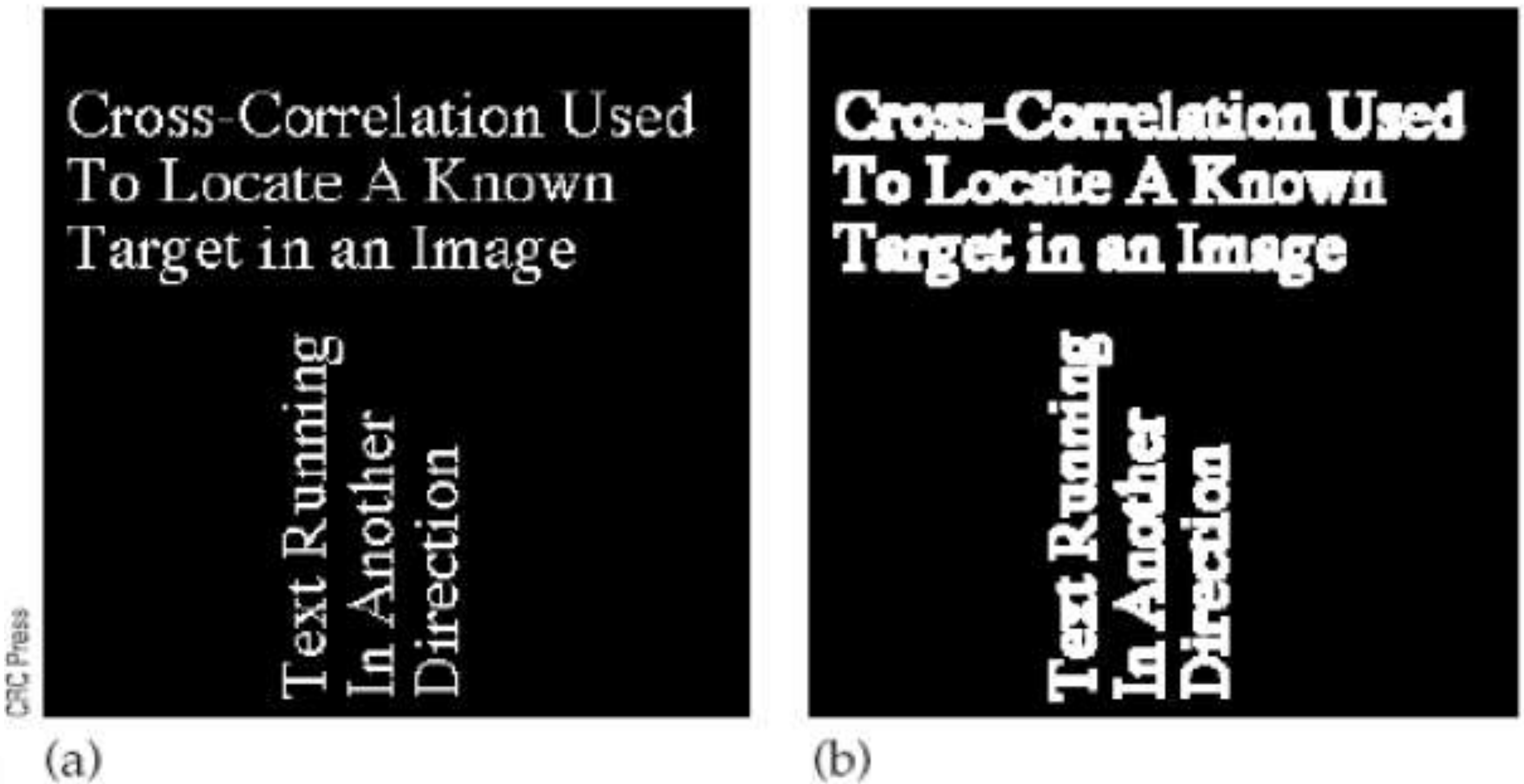


그림 10.5 이진 영상의 팽창 (a) Text 영상 (b) 팽창 결과

10.3.2 침식(Erosion) 영상 내 객체 인식을 위하여 불필요한 부분을 제거한다.

주어진 집합 A와 B에 대하여 B에 의한 A의 침식은 A⊖B로 표시하며 아래와 같이 정의된다. (0,0) point 가 포함되어 B가 A에 완전히 포함되도록 배치

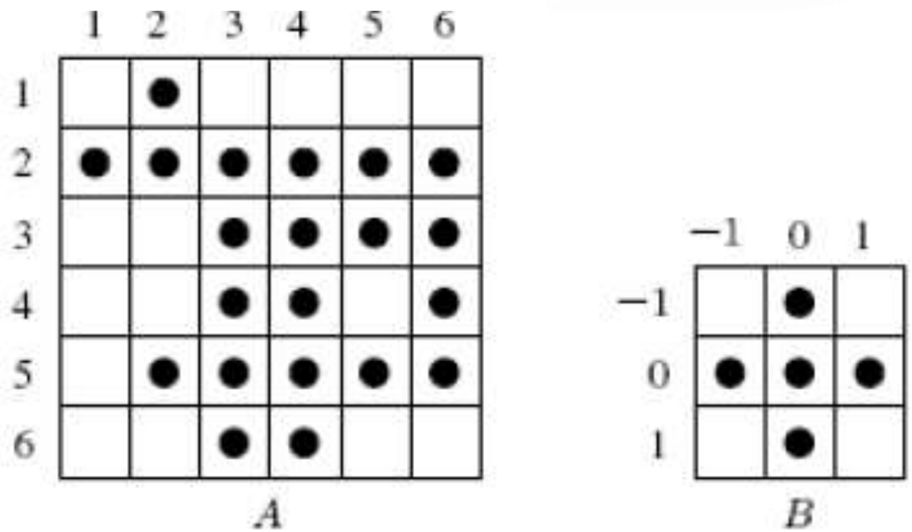
$$A \ominus B = \{w : B_w \subseteq A\}.$$

바꾸어 말하면 B에 의한 A의 침식은, B_w가 A의 내부에서 w=(x,y)의 모든 점으로 구성된다. 침식을 실행하기 위하여 B를 A에 걸쳐서 이동시키면서 B가 A의 내부에 완전히 속할 때 B의 (0,0)점에 대응하는 점을 표시(마크)한다. 이렇게 얻어진 모든 점들의 집합을 구하면 침식의 결과이다.

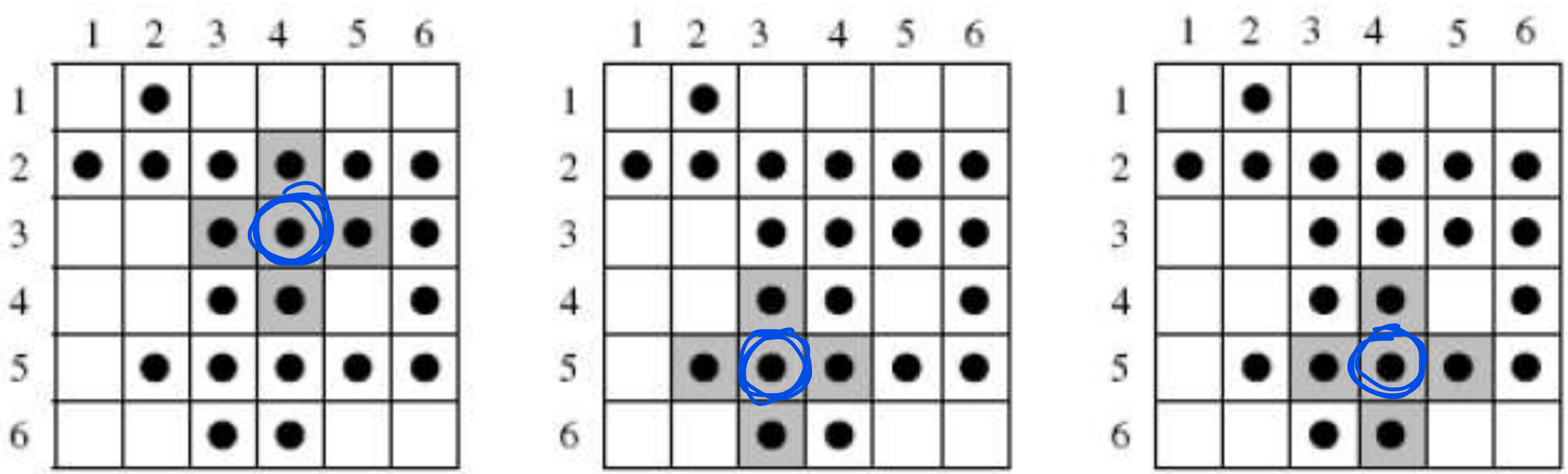
침식의 예는 그림 10.6에 보였다. 이 예에서 침식A⊖B는 A의 부분집합이 된다. 이것은 반드시 그럴 필요는 없다. B의 원점의 위치에 따라 다르다. 만일 B가 원점을 포함한다면(그림 10.6), 침식은 원 물체의 부분집합이 된다.

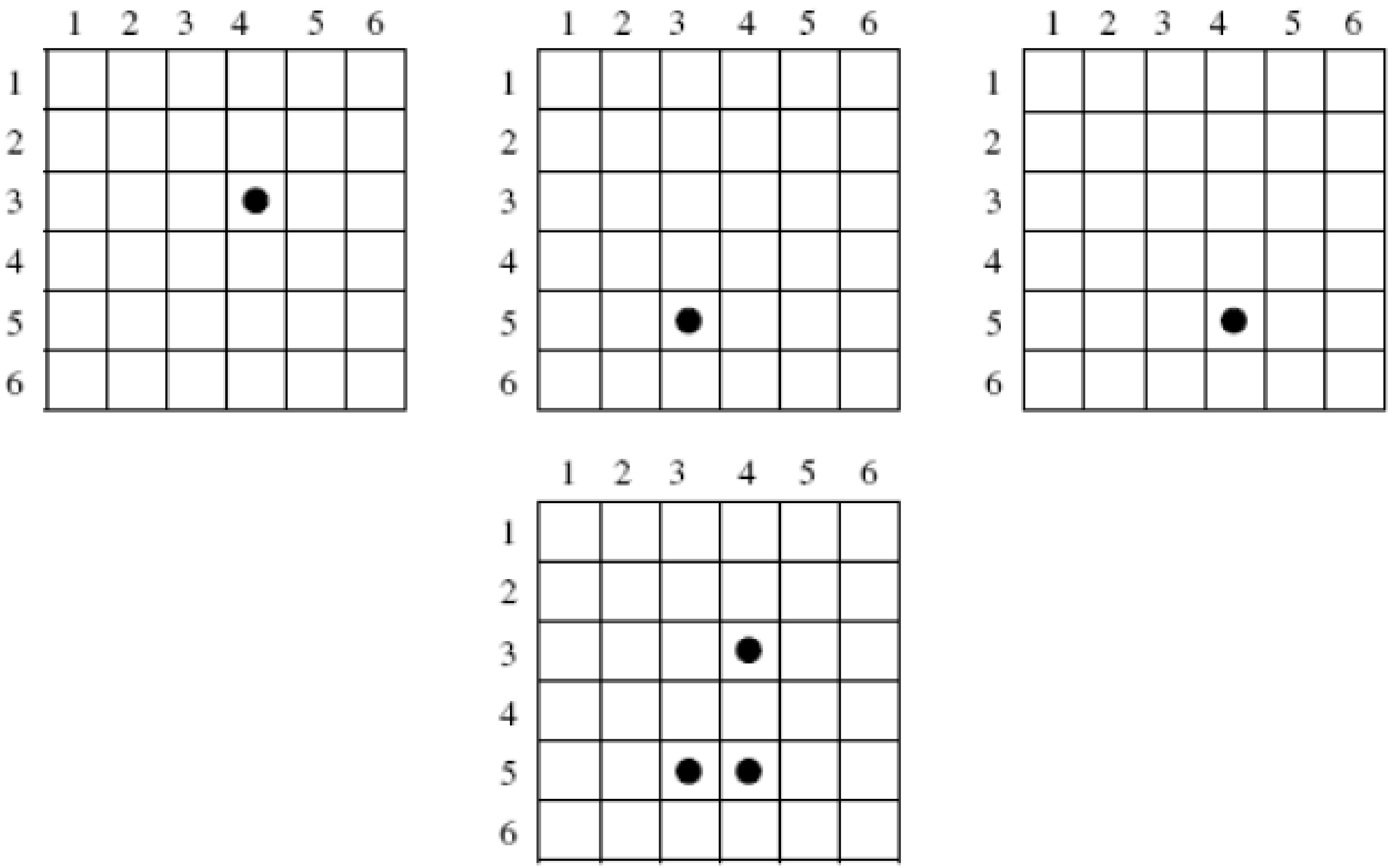
그림 10.7은 B가 원점을 포함하지 않는 예이다. 그림 10.7 (b)에서 흰 원이 침식의 결과이다.

B가 A에 완전히 포함되는 곳 찾기



+ : 크로스 마스크





$A \oplus B$

그림 10.6 교차형 구조의 침식연산 과정과 그 결과

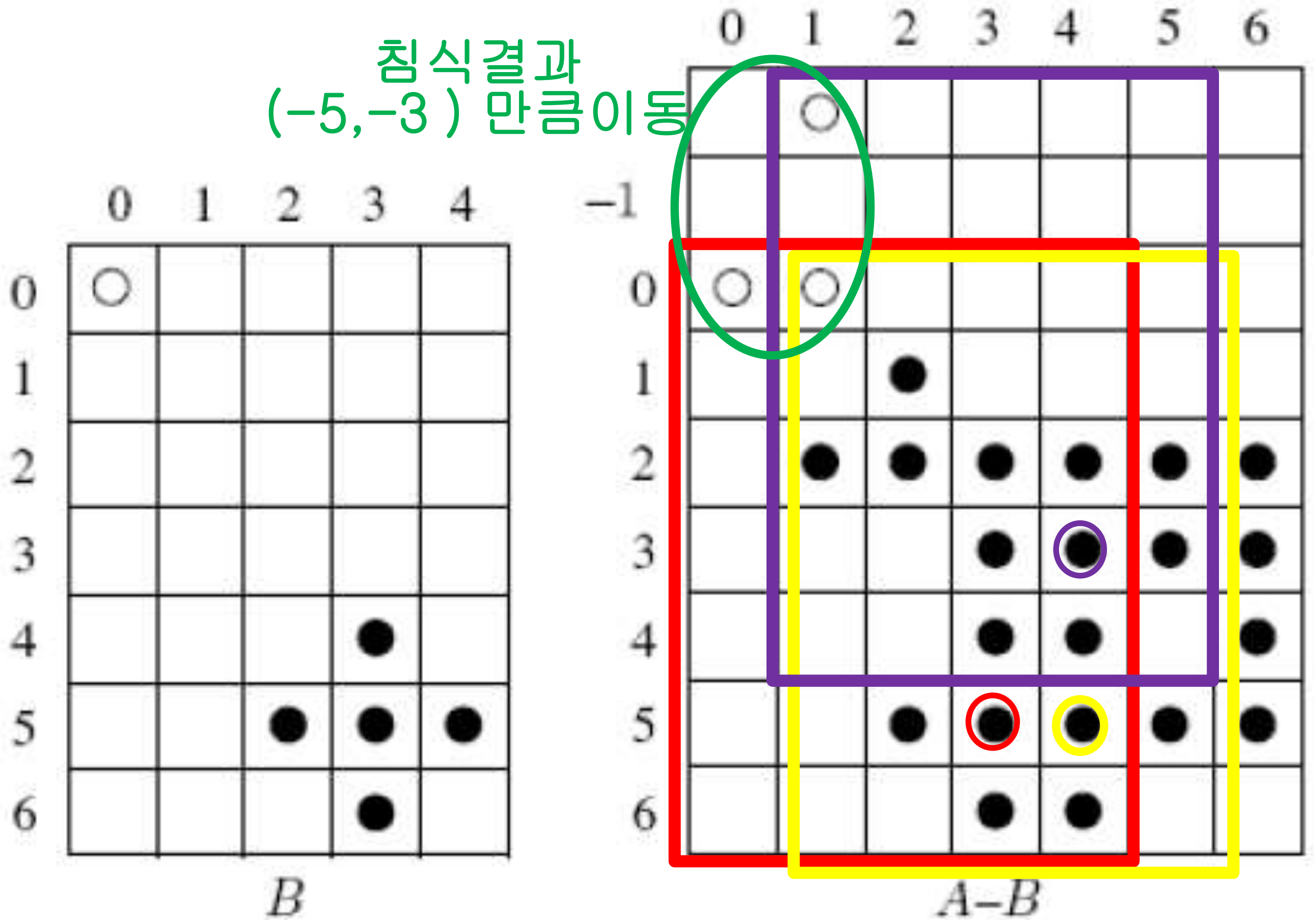


그림 10.7 원점을 포함하지 않는 구조(B)에 대한 침식연산 결과

그림 10.7에서 침식의 모양은 그림 10.6에서와 같다. 그러나 그 위치는 다르다. 그림 10.7에서 B의 원점이 그림 10.6의 위치에서 (−5,−3)만큼 이동되어 있다. 우리는 침식이 그 해당 양만큼 이동된다는 것을 알 수 있다. 그림 10.6과 10.7을 비교하면, 2번째 침식은 처음보다 (−5,−3)만큼 실제로 이동됨을 볼 수 있다.

팽창과 마찬가지로 침식에 대하여, 일반적으로 A는 처리될 영상이고, B는 화소의 작은 집합으로 구조적 요소 혹은 커널(kernel)에 해당 한다.

침식은 Minkowski subtraction에 관련되고, 아래와 같이 정의 된다.

$$A \ominus B = \bigcap_{b \in B} A_b.$$

MATLAB에서 침식은 다음의 명령으로 실행할 수 있다.

input mask

```
>> imerode(image,kernel)
```

하나의 예로서, 하나의 다른 이진 영상을 이용하여 아래와 같이 적용한다.

```
>> c=imread('circbw.tif');
>> ce=imerode(c,sq);
>> subplot(1,2,1),imshow(c)
>> subplot(1,2,2),imshow(ce)
```

이 결과는 그림 10.8 (b)에 보였다. 여기서 영상이 얇아짐을 알 수 있다. 이것은 말 그대로 침식의 결과를 준다. 만일 영상을 계속하여 침식을 적용하면 결국에는 완전히 빈 영상으로 남을 것이다.

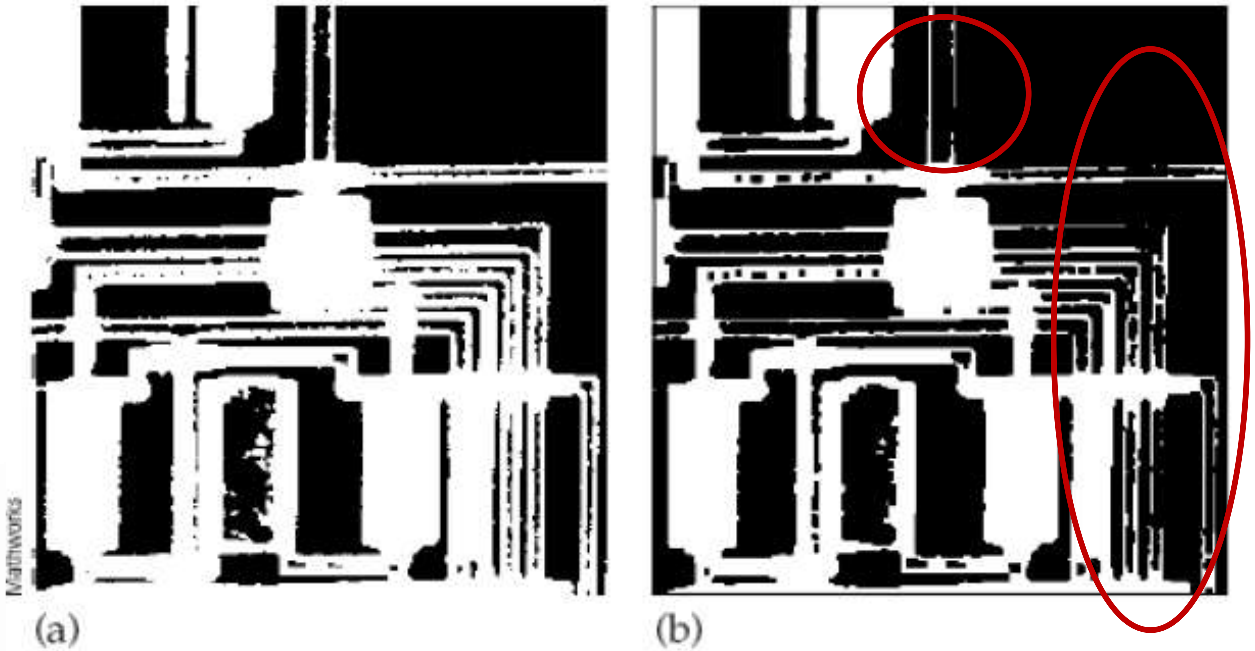


그림 10.8 이진 영상의 침식연산 (a) 원 영상 (b) 침식결과

침식과 팽창과의 관계

침식과 팽창이 서로 반대의 연산임을 알 수 있다. 특히 침식에 대한 전체의 보수(complement)는 아래와 같이 각 보수(complement)에 대한 팽창과 등가이다.

$$\overline{A \ominus B} = \overline{A} \oplus B.$$

이의 증명은 전문서적을 참조하기 바란다.

이와 유사하게 침식과 팽창을 서로 교환하여도 아래와 같이 성립한다.

$$\overline{A \oplus B} = \overline{A} \ominus B.$$

MATLAB 명령을 이용하여 이것이 성립함을 증명할 수 있다. 여기서 알아야 할 것은, 이진 영상 b의 보수이고 아래와 같이 얻는다.

```
>> ~b
```

그리고 주어진 2개의 영상 a와 b에서 그들이 등가임을 아래와 같이 결정한다.

```
>> all(a(:)==b(:))
```

$$\overline{A \ominus B} = \overline{A} \oplus \hat{B},$$

이 식이 등가임을 증명하기 위해 text 영상과 구조적 요소(커널)을 지정한다. 방정식의 왼쪽에 대하여 아래와 같이 표현하고,

```
>> lhs=~imerode(t,sq);
```

식의 오른쪽에 대하여는 다음과 같이 적용하여,

```
>> rhs=imdilate(~t,sq);
```

최종적으로 아래의 명령으로 1을 return하면 구해진다..

```
>> all(lhs(:)==rhs(:))
```

팽창과 침식을 이용한 Edge 검출

10.3.3 응용 : 경계선 검출(Boundary detection)

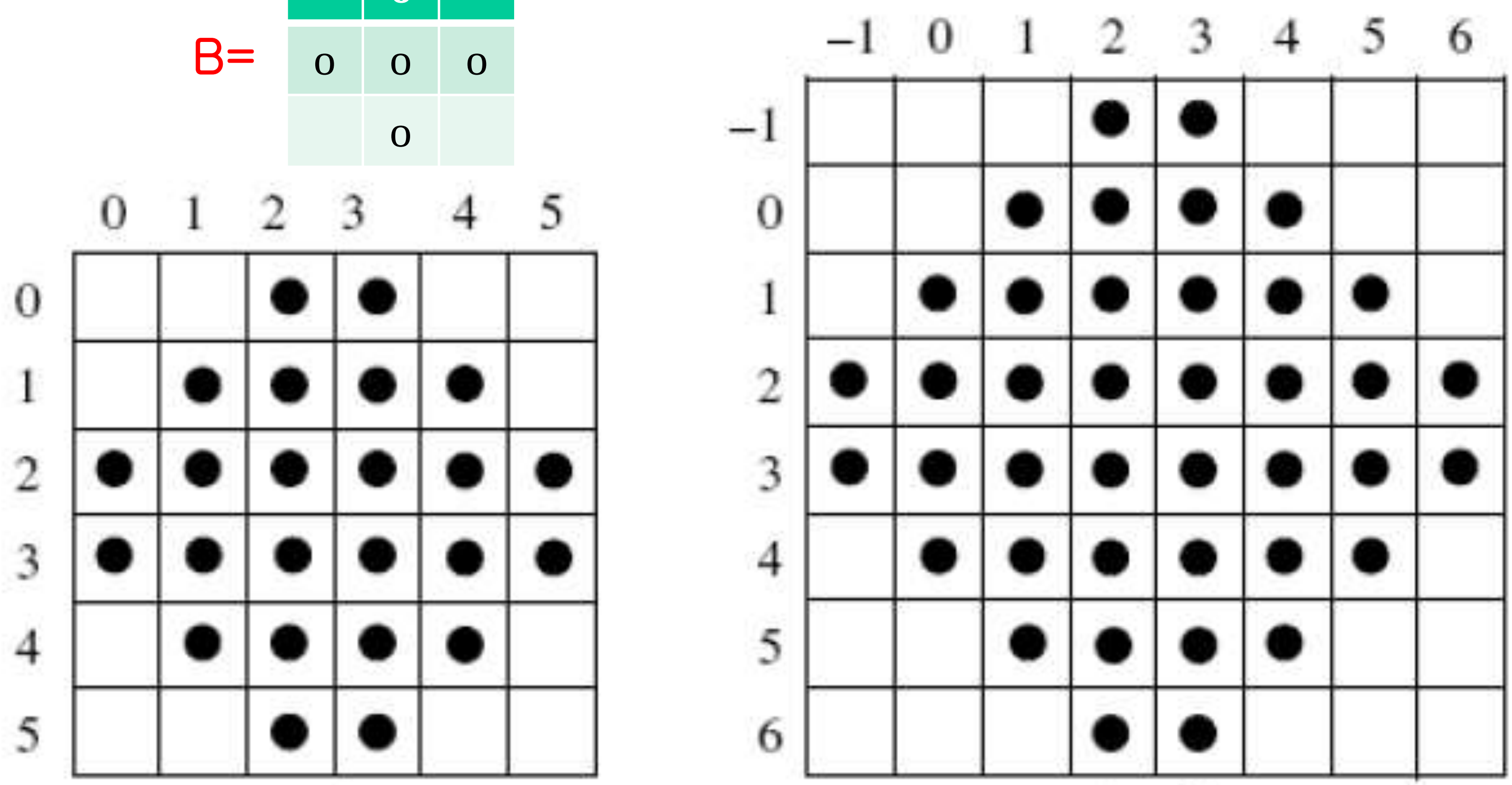
A는 영상이고, B는 원점에 대칭적인 점들로 구성되는 구조적 요소(커널)이라고 하면, 다음과 같은 방법으로 A의 경계를 정의할 수 있다.

(i) $A - (A \ominus B)$ internal boundary 원본과 크기 같음
(ii) $(A \oplus B) - A$ external boundary 원본보다 큼
(iii) $(A \oplus B) - (A \ominus B)$ morphological gradient 커지고 두꺼워짐

각 정의에서 − 부호는 집합의 차분에 해당한다. 그림 10.9와 같이, 몇 가지의 예에서 내부 경계는 A의 내부에 있는 화소들로 구성되는데 이들은 에지가 되고, 외부의 경계는 A의 바깥쪽 화소들로서 내부 경계와 바로 이웃하고 있다. 형태학적 기울기는 내부 및 외부 경계의 결합이다.

B=

	o	
o	o	o
	o	



A $A \oplus B$

팽창

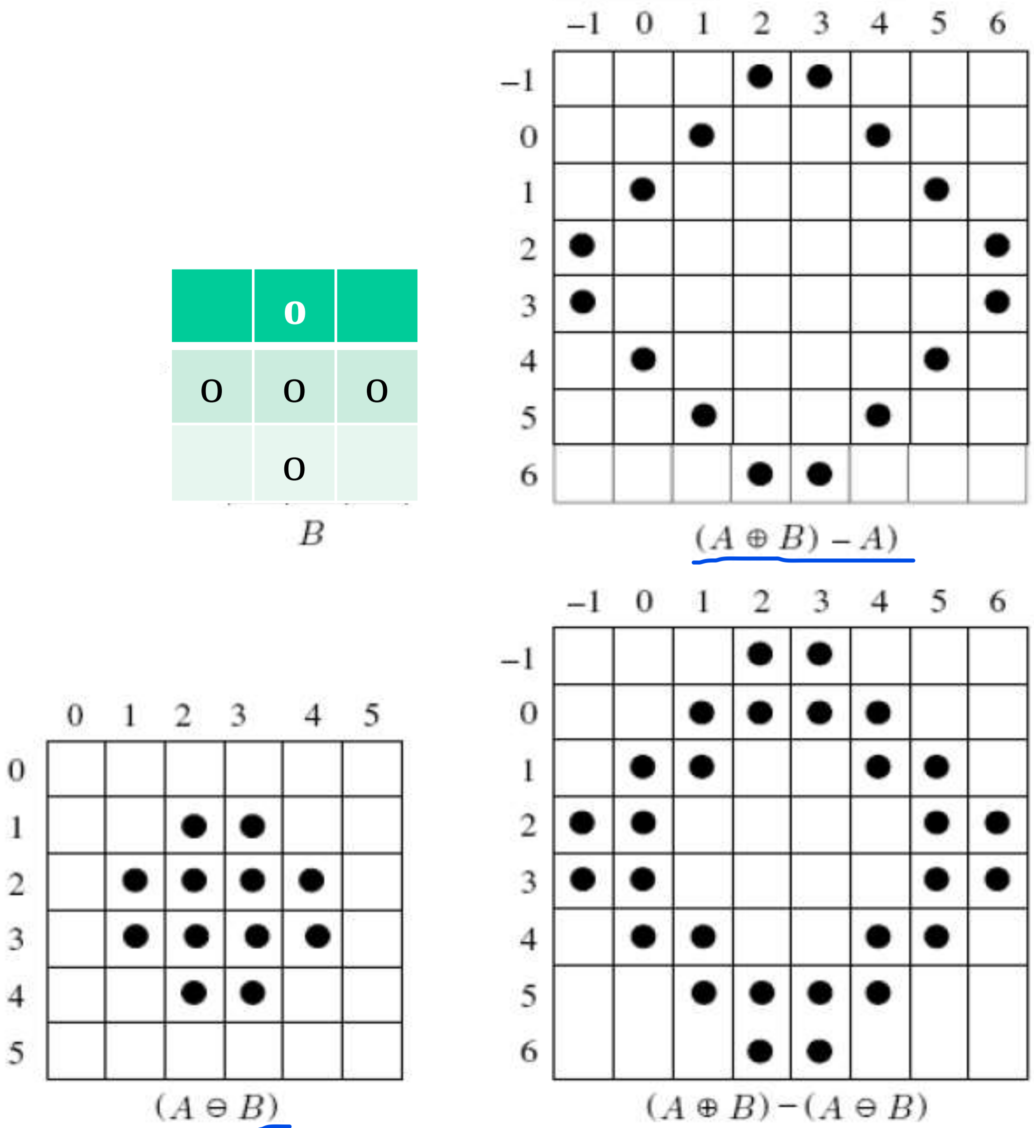


그림 10.9 경계 추출

몇 가지 예를 보기 위해 영상을 rice.tif를 선택하고 이진 영상을 얻기 위해 아래와 같이 문턱치 처리를 한다.

```
>> rice=imread('rice.tif');
>> r=rice>110;  110 보다 큰 것은 백색으로
```

다음으로 내부 경계를 얻기 위해 아래와 같이 처리한다.

```
>> re=imerode(r,sq);
>> r_int=r&~re;    % r-re와 동일
>> subplot(1,2,1),imshow(r)
>> subplot(1,2,2),imshow(r_int)
```

이 결과는 그림 10.10 (b)와 같다. 유사하게 외부 경계와 형태학적 기울기는 아래와 같이 얻는다.

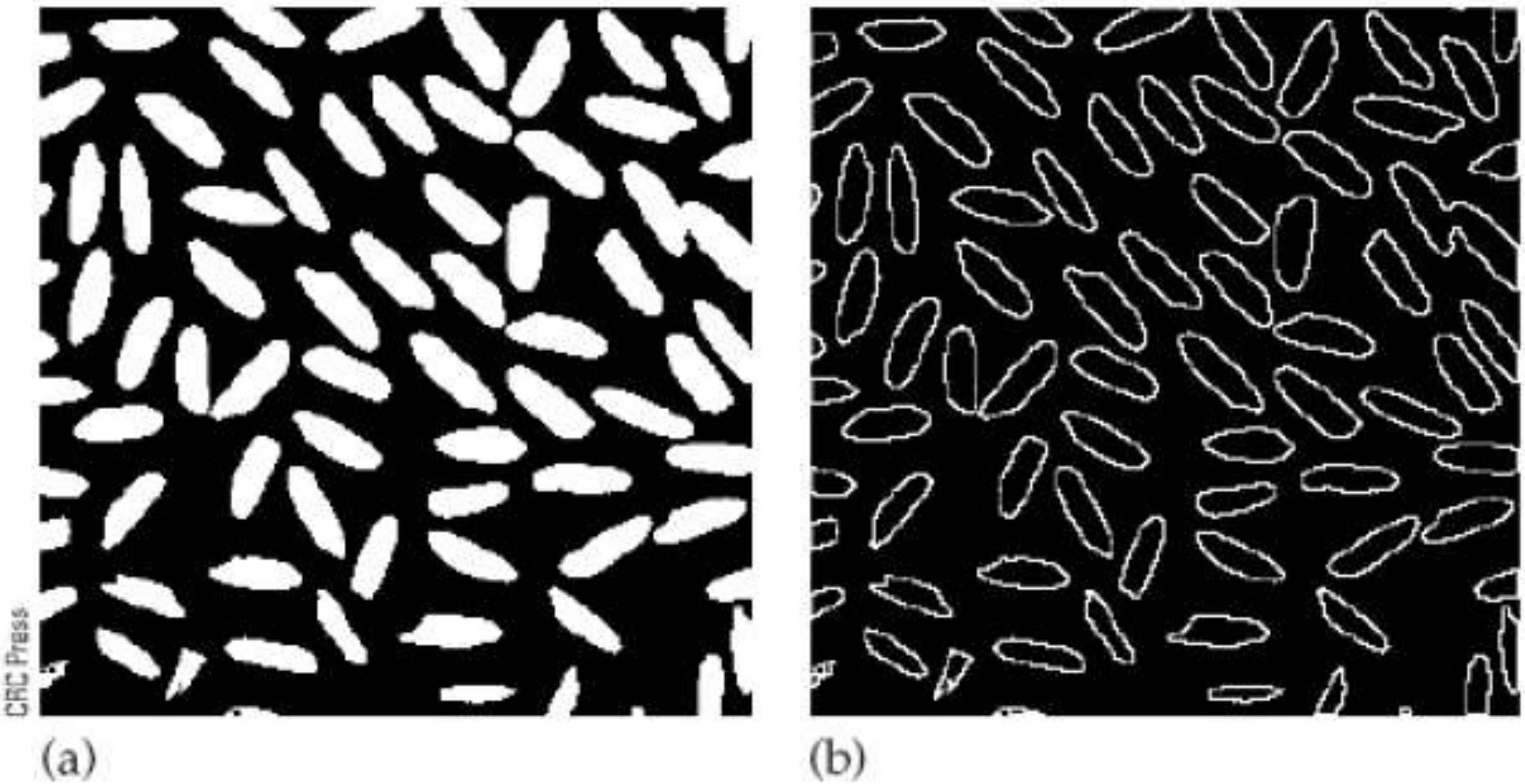
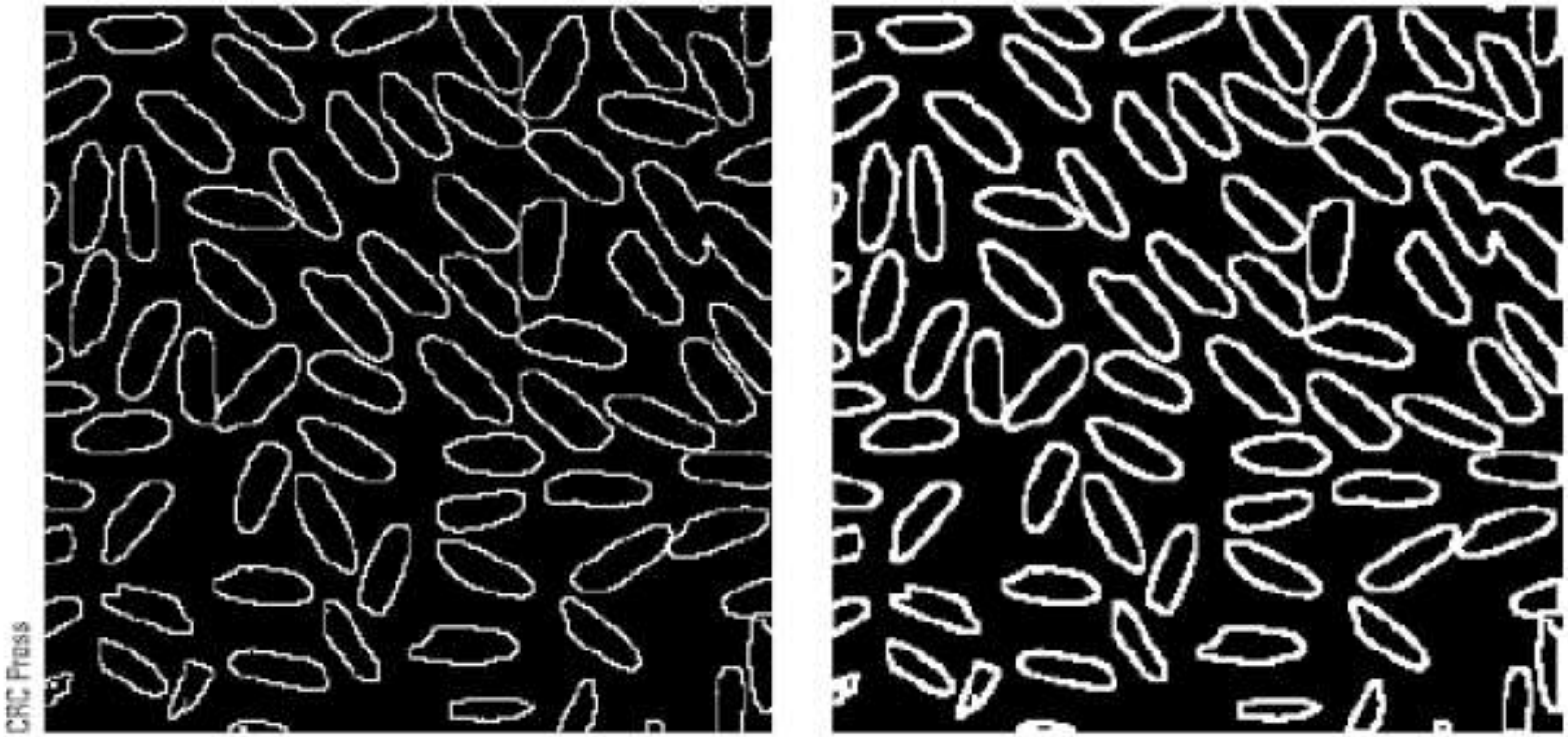


그림 10.10 형태학적 에지검출 (a) rice 영상 (b) 내부 경계

```
>> rd=imdilate(r,sq);
>> r_ext=rd&~r;    % rd-r과 동일
>> r_grad=rd&~re;  % rd-re와 동일
>> subplot(1,2,1),imshow(r_ext)
>> subplot(1,2,2),imshow(r_grad)
```

이 결과는 그림 10.11과 같다.

외부 경계는 내부의 경계보다 더 크다. 왜냐하면 내부의 경계는 영상 물체의 외면을 나타내고, 이에 비해서 외부의 경계는 영상의 물체 부분이 아닌 바깥쪽의 화소로 에지와 이웃하는 배경 부분이기 때문이다. 형태학적 기울기는 실제로 이들의 결합으로서 두텁게 표현된다.



(a) % rd-r (b) % rd-re

그림 10.11 형태학적 에지검출 (a) 외부 경계 (b) 형태학적 기울기

10.4 열림(Opening)과 닫힘(Closing)

이들 연산은 팽창과 침식의 기본 연산으로 2차적 연산으로 실행한다. 이들은 또한 수학적으로 더 좋은 특성을 가짐을 알 수 있다.

10.4.1 열림 연산(Opening)

주어진 A와 구조적 커널 B에 대하여 B에 의한 A의 열림 연산은 AoB로 표기하고, 아래와 같이 정의 한다.

$$A \circ B = (A \ominus B) \oplus B.$$

따라서 침식연산 후에 팽창 연산으로 구성된다. 그 등가 표현으로 아래와 같이 표현할 수 있다.

$$A \circ B = \cup\{B_w : B_w \subseteq A\}.$$

즉, AoB는, 완전히 A의 내부에 포개지는 모든 B의 이동을 조합하는 것이다. 침식연산과의 차이점은 침식연산은 B를 이동하면서 A의 내부에 완전히 포개지는 상태에서 B의 (0,0)점에서만 구성되는 것이지만, 열림 연산은 (0,0)점을 포함하고 또 좌우 및 아래 위에 있는 B의 성분으로 이동시킨 A의 전체 조합하는 것이다. 그림 10.12에 이 열림 연산의 예를 보였다.

B=

	o	
o	o	o
	o	

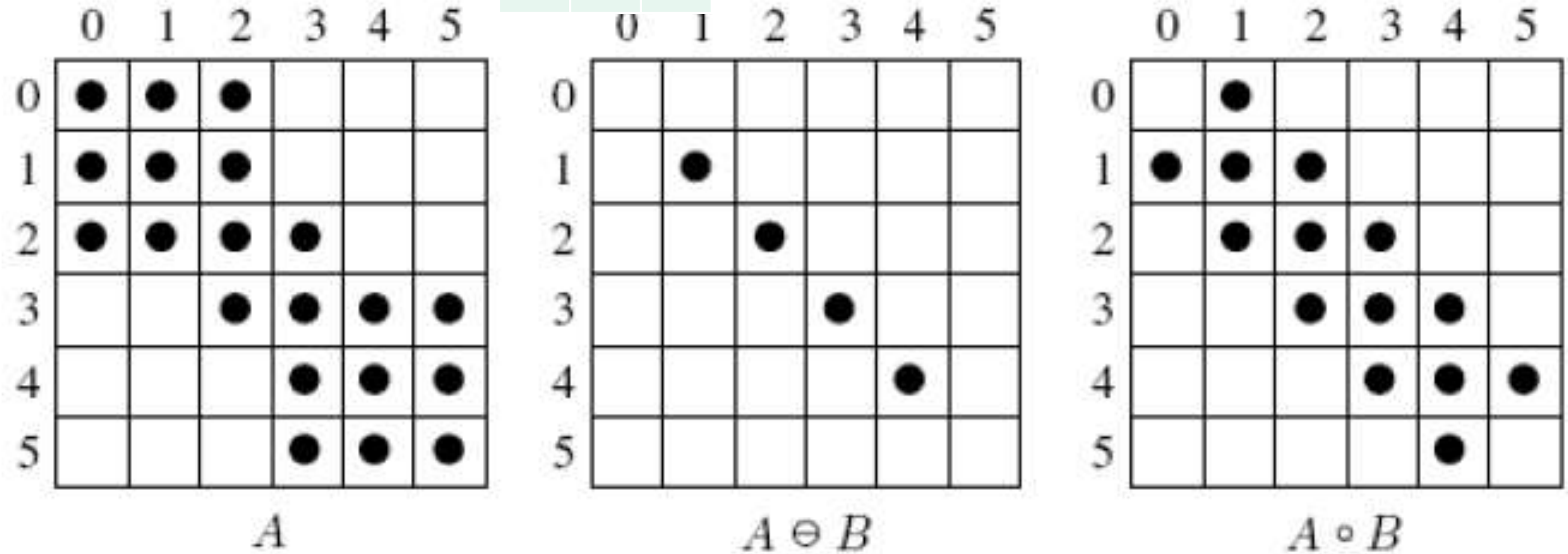


그림 10.12 열림 연산의 예

열림 연산은 다음과 같은 성질을 만족한다.

1. $(A \circ B) \subseteq A$. 이는 침식연산의 경우와 다르다. 앞에서와 같이 침식은 부분집합일 필요는 없다.
2. $(A \circ B) \circ B = A \circ B$. 즉, 열림 연산은 1회 이상 처리할 수 필요가 없다. 이 성질은 idempotence(제곱한 것과 같은 값)라는 특성을 가진다. 이러한 점이 침식과 다르다. 침식연산은 연속 적용하여 영상이 없어질 때까지 반복 적용할 수 있다.
3. $A \subseteq C$ 이면, $(A \circ B) \subseteq (C \circ B)$ 이다.
4. 열림 연산은 영상을 스므딩하는 경향이 있고, 좁은 연결점을 끊으며, 돌출부분을 제거하는 성질을 가진다.

10.4.2 닫힘 연산(Closing)

열림 연산과 유사하게 닫힘 연산은 팽창 연산 후에 침식연산으로 이루어지며, A•B로 표시하고 아래와 같이 정의한다.

$$A \bullet B = (A \oplus B) \ominus B.$$

닫힘 연산의 또 다른 정의는, x를 포함하는 모든 이동성분 B_w가 A와의 교집합에서 공집합(원소가 없는 집합)이 아니면 x∈(A•B) 이다. 그림 10.13은 닫힘 연산의 예를 나타낸다. 닫힘 연산은 다음과 같은 성질을 가진다.

1. A⊆ (A•B).
2. (A•B)•B = A•B 이다. 즉, 닫힘 연산은 열림 연산과 같이 idempotence(제곱한 것 과 같은 값)이다.
3. A⊆C 이면, (A•B)⊆(C•B) 이다.
4. 닫힘 연산은 영상을 스므딩하지만, 좁은 연결점을 융합하고 갈라진 틈을 좁히며 작은 홀을 제거한다.

열림 연산과 닫힘 연산은 imopen과 imclose 함수로 각각 구현한다. 단순한 영상에 대하여 정방형 및 교차형 구조적 커널을 이용하여 그 효과를 볼 수 있다.

B=

	0	
0	0	0
	0	

영상 A를 (-1,0),(0,-1),(0,0),(0,1),(-1,0) 만큼 이동한 합집합

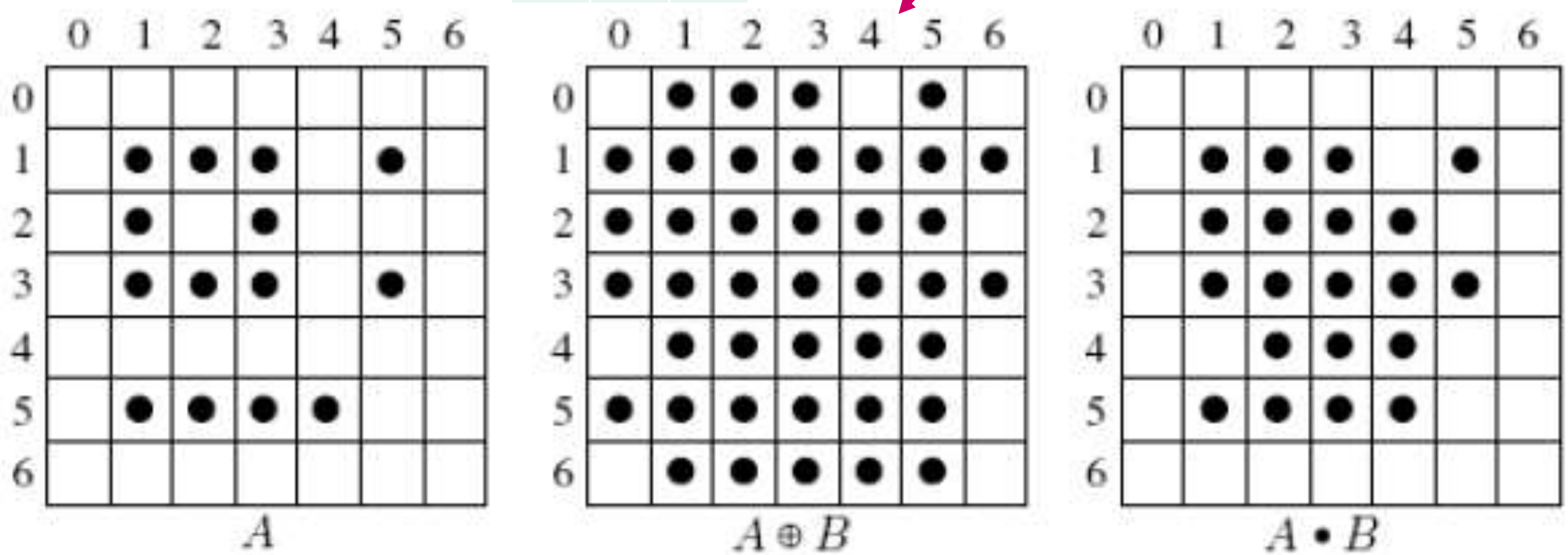


그림 10.13 닫힘 연산 과정 및 그 결과

```
>> cr=[0 1 0;1 1 1;0 1 0];
>> >> test=zeros(10,10);test(2:6,2:4)=1;test(3:5,6:9)=1;
      test(8:9,4:8)=1;test(4,5)=1
test =
     0     0     0     0     0     0     0     0     0     0
     0     1     1     1     0     0     0     0     0     0
     0     1     1     1     0     1     1     1     1     0
     0     1     1     1     1     1     1     1     1     0
     0     1     1     1     0     1     1     1     1     0
     0     1     1     1     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     1     1     1     1     1     0     0
     0     0     0     1     1     1     1     1     0     0
     0     0     0     0     0     0     0     0     0     0
>> imopen(test,sq)
ans =
     0     0     0     0     0     0     0     0     0     0
     0     1     1     1     0     0     0     0     0     0
     0     1     1     1     0     1     1     1     1     0
     0     1     1     1     0     1     1     1     1     0
     0     1     1     1     0     1     1     1     1     0
     0     1     1     1     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
```

끊어짐

제거됨

Sq=

1	1	1
1	1	1
1	1	1

Cr=

	1	
1	1	1
	1	

```
>> imopen(test,cr)
ans =
     0     0     0     0     0     0     0     0     0     0
     0     0     1     0     0     0     0     0     0     0
     0     1     1     1     0     1     1     1     0     0
     0     1     1     1     1     1     1     1     1     0
     0     1     1     1     0     1     1     1     0     0
     0     0     1     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
```

대각선을 넘지 못한다.

각 경우에서 영상은 성분들로 분리되고 아래쪽 부분은 완전히 제거됨을 알 수 있다.

닫힘에서는 영상이 완전히 연결되는데, 대각 구조의 커널을 이용하여 text 영상으로 연결되는 효과를 얻을 수 있다.

```
>> imclose(test,sq)

ans =

     1     1     1     1     0     0     0     0     0     0
     1     1     1     1     0     0     0     0     0     0
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     0     0
     0     0     0     1     1     1     1     1     0     0
     0     0     0     1     1     1     1     1     0     0
     0     0     0     1     1     1     1     1     0     0
     0     0     0     1     1     1     1     1     0     0
```

```
>> imclose(test,cr)
```

대각선에 취약함

```
ans =

     0     0     1     0     0     0     0     0     0     0
     0     1     1     1     0     0     0     0     0     0
     1     1     1     1     1     1     1     1     1     0
     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     0
     0     1     1     1     1     1     1     1     0     0
     0     0     1     1     1     1     1     0     0     0
     0     0     0     1     1     1     1     1     0     0
     0     0     0     1     1     1     1     1     0     0
     0     0     0     0     1     1     1     0     0     0
```

```
>> diag=[0 0 1;0 1 0;1 0 0]

diag =

     0     0     1
     0     1     0
     1     0     0

>> tc=imclose(t,diag);
>> imshow(tc)
```

이 결과는 그림 10.14와 같다.

대각선으로 연결된 부분이 보인다.

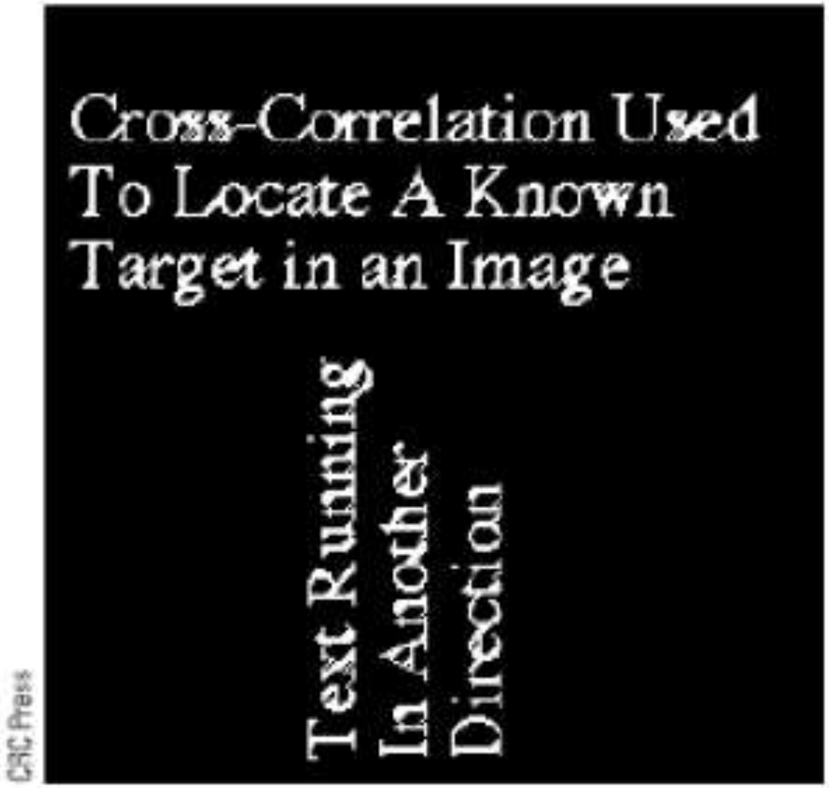


그림 10.14 닫힘 연산의 예

응용 : 잡음제거

A가 임펄스잡음이 첨가된 이진 영상(흰색 화소의 일부가 흑색이고, 흑색 화소의 일부가 흰색으로 나타남)이라고 가정하자. 그림 10.15를 참조할 것.

A⊖B는 한 점의 흑색 화소를 제거하지만, 구멍을 크게 한다. 이를 아래와 같이 2회의 팽창 연산으로 채울 수 있다.

$((A \circ B) \bullet B)$. 노이즈의 크기를 작게하고 백그라운드를 밀고들어오게 함

이것이 소위 **형태학적 필터링**이다.

하나의 영상을 택하고, 여기에 10%의 shot잡음을 아래와 같이 가한다.

```
>> c=imread('circles.tif');
>> x=rand(size(c));
>> d1=find(x<=0.05);
>> d2=find(x>=0.95);
>> c(d1)=0;
>> c(d2)=1;
>> imshow(c)
```

이 결과가 그림 10.15 (a)에 보였다. 필터링 과정은 아래와 같이 구현할 수 있다.

```
>> cf1=imclose(imopen(c,sq),sq);
>> figure,imshow(cf1)
>> cf2=imclose(imopen(c,cr),cr);
>> figure,imshow(cf2)
```

이 결과는 그림 10.15 (b)와 (c)에 보였다. 교차형 구조 커널에서 더욱 적게 나타나지만, 결과에 약간의 블록화 현상을 볼 수 있다.

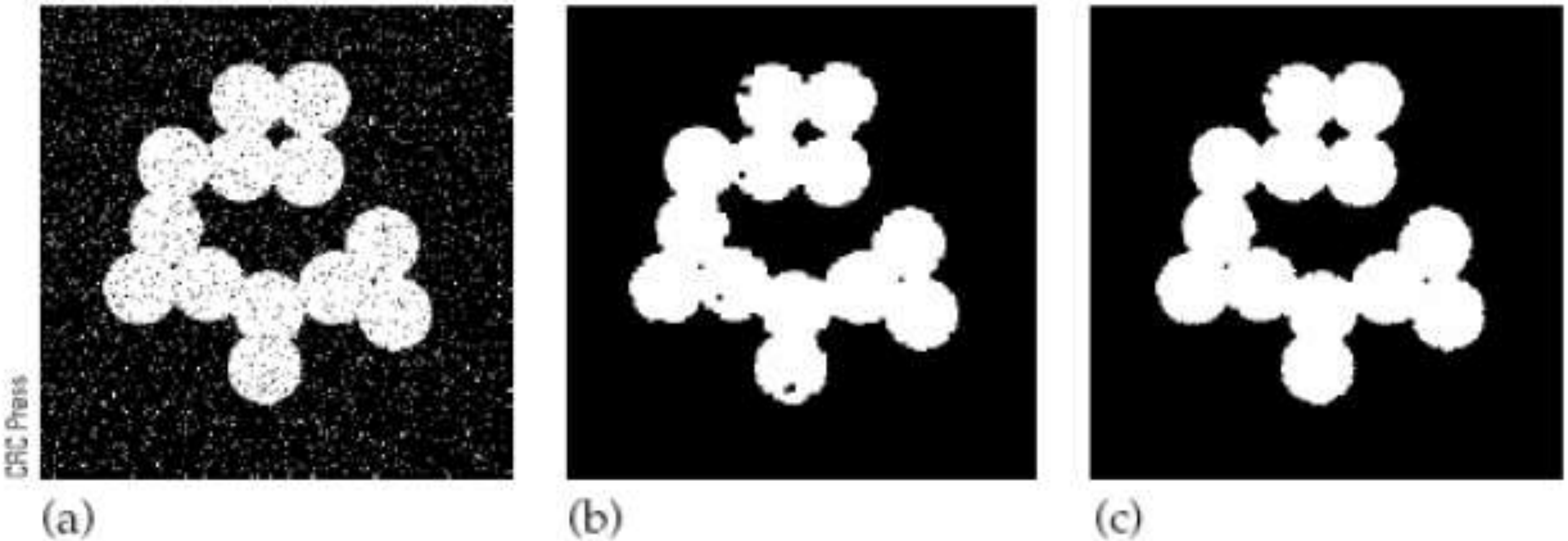


그림 10.15 잡음이 있는 이진 영상과 형태학적 필터링을 적용한 결과 (a)이진 영상 (b)구형 커널 적용 결과 (c)교차형 커널을 적용한 결과

열림과 닫힘 연산의 관계

열림 연산과 닫힘 연산은 침식연산과 팽창 연산의 관계와 유사한 관계를 공유한다. 열림의 전체에 대한 보수(complement)는 각 보수의 닫힘 연산과 같고, 닫힘에 대한 전체의 보수는 각 보수의 열림 연산과 아래와 같이 서로 같다. 특히

$$\overline{A \bullet B} = \overline{A} \circ B$$

$$\overline{A \circ B} = \overline{A} \bullet B.$$

10.5 Hit-or-Miss 변환

찾고자하는 object를 찾는법

Hit-or-Miss 변환은 영상에서 형태를 찾기 위한 하나의 강력한 방법이다. 다른 모든 형태학적 알고리즘과 마찬가지로, 전적으로 팽창과 침식연산으로 정의될 수 있고, 이 경우에는 침식으로만 정의될 수 있다.

그림 10.16에서 영상 A의 중심에 3X3 정방형 모양을 위치시킨다고 가정한다. 정방형 구조적 커널인 B로서 침식 연산 A⊖B를 실행한다면 그림 10.17의 결과를 얻게 된다. 이 결과는 2개의 화소를 포함한다. 왜냐하면 A에 B를 이동시켜서 완전히 포개지는 위치가 2곳 밖에 없기 때문이다. 또 구조적 커널 C를 A의 보수에 침식시킨다고 가정한다. 여기서 C는 정방형 3X3의 외곽형 구조적 커널이다. $\bar{A}$와 C는 그림 10.18과 같다.[C의 중심은 (0,0)이라 가정]

침식 연산 $\bar{A}$⊖C을 실행하면 그림 10.19의 결과를 얻는다.

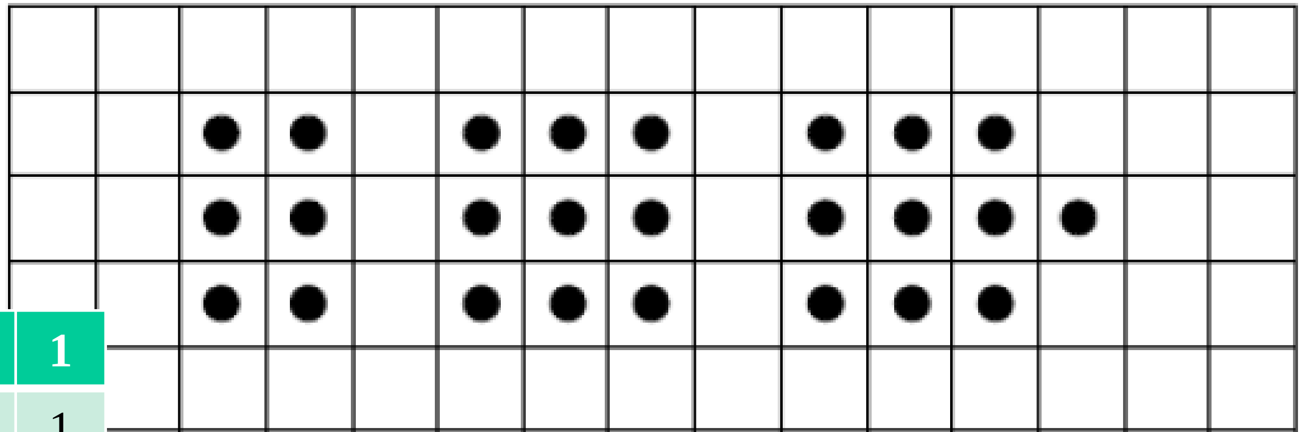


그림 10.16 얻어야 할 형상을 포함한 영상 A

B=

1	1	1
1	1	1
1	1	1

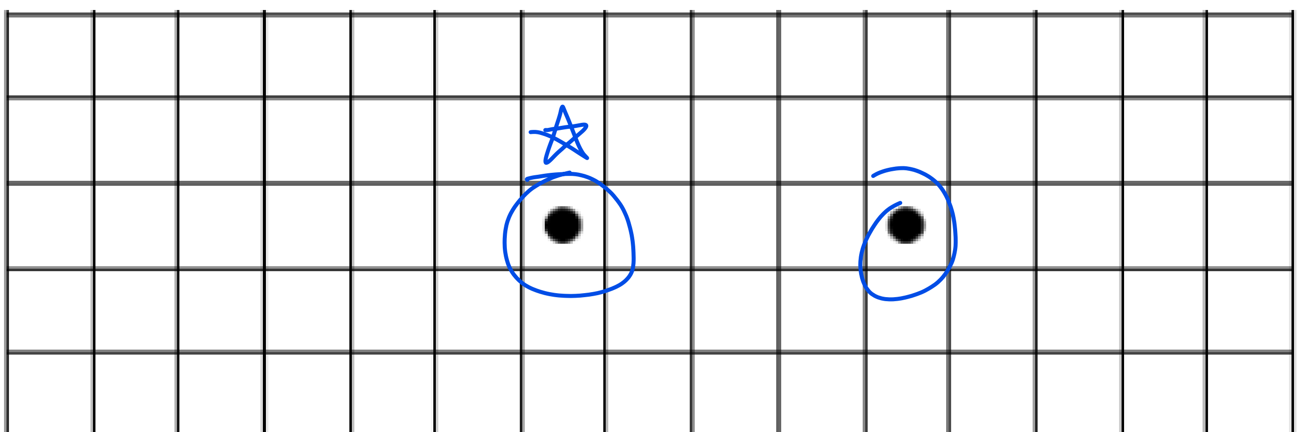


그림 10.17 침식 연산 A⊖B의 결과

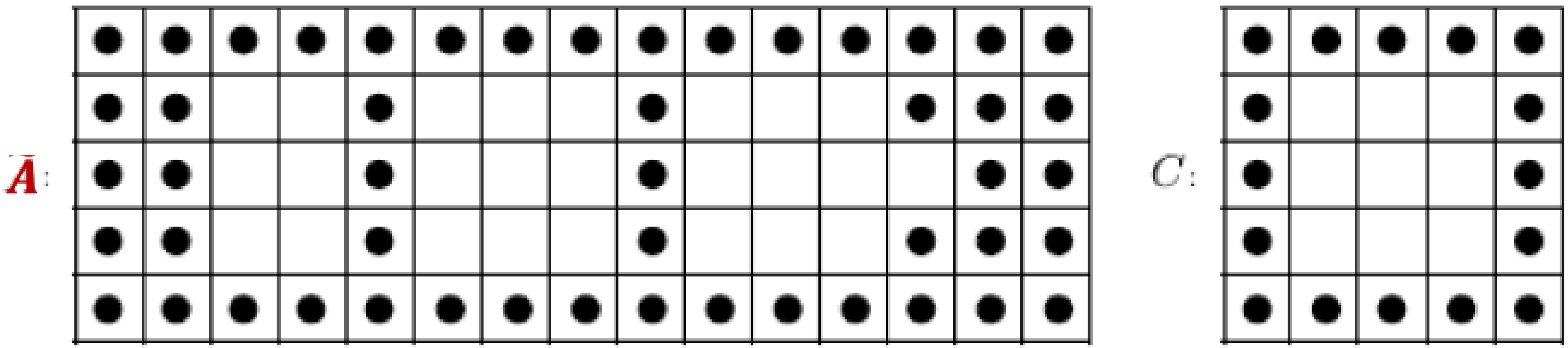


그림 10.18 영상 Ā의 보수와 2번째 구조적 커널 C

마스크 B를 보수

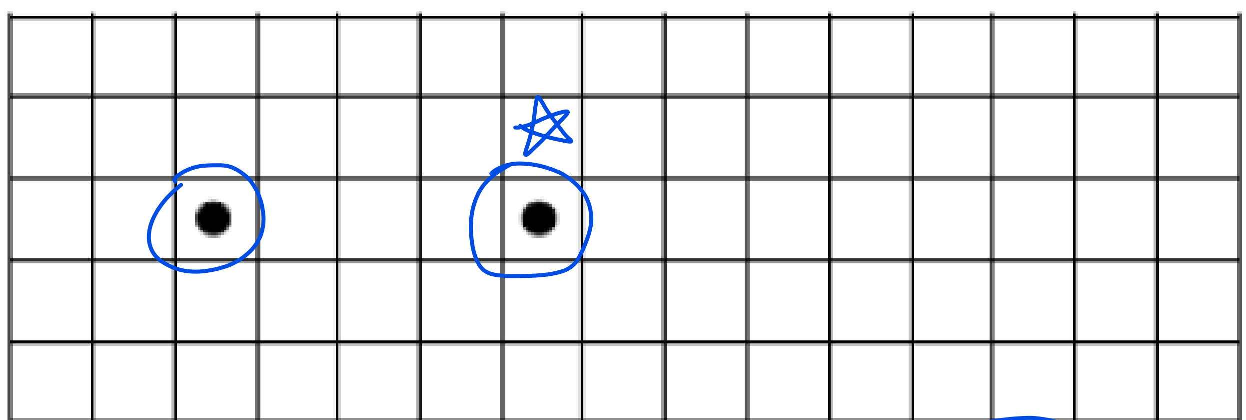


그림 10.19 침식 연산 $\bar{A} \ominus C$의 결과

2개의 침식 연산의 교집합은 A에서 3X3 정방형의 중심의 위치에서 단 1개의 화소가 구해진다. 이것이 우리가 원하는 바로 그것이다. 만일 A가 1개 이상의 정방형을 포함한다면 최종 결과는 각 중심의 위치에 1개의 화소가 얻어질 것이다. 이 침식들의 결합이 hit-or-miss 변환이다.

일반적으로 영상에서 특별한 형상을 찾고자 한다면 2개의 구조적 커널을 고안해야 하는데, B_1은 동일한 모양이고, B_2는 외곽형 커널이다. hit-or-miss 변환을 $B=(B_1,B_2)$

$$A \circledast B = (A \ominus B_1) \cap (\overline{A} \ominus B_2)$$

와 아래의 식으로 쓸 수 있다.

예를 들면, 그림 10.5의 text 영상에서, “Cross-Correlation"에 있는 - (hyphen)을 구해보기로 한다. 사실 이것은 화소길이가 6인 라인이다. 그래서 2개의 구조적 커널을 아래와 같이 구한다.

```
>> b1=ones(1,6);
>> b2=[1 1 1 1 1 1 1 1;1 0 0 0 0 0 0 1; 1 1 1 1 1 1 1 1];
>> tb1=imerode(t, b1);
>> tb2=imerode(~t, b2);
>> hit_or_miss=tb1&tb2;
>> [x,y]=find(hit_or_miss==1)
```

구 후에 이를 (41,76)의 좌표로 return 한다. 이 좌표는 hyphen의 중간이 적당하다. 아래의 명령은 충분하지는 않다. 왜냐하면 이 영상에서 길이가 6인 직선이 여러 개가 존재하기 때문이다.

```
>> tb1=imerode(t, b1);
```

그림 10.20에 주어진 영상 tb1을 보면 이를 알 수 있다.

그림 10.20 hyphen 모양의 구조적 커널에 의한 침식된 text 영상

10.6 여러 가지의 형태적 알고리즘

이 절에서는 앞 절에서 논의한 몇 가지 형태학적 기술을 이용한 dfj 가지 간단한 알고리즘들을 소개하기로 한다.

10.6.1 영역 채우기(Region filling)

영상에서 8-연결 경계에 의해 그림 10.21과 같이 영역경계를 가진다고 가정한다. 그 영역 내에 화소 p가 주어지면, 전체 영역에 걸쳐서 채우기를 한다. 이렇게 하기 위해, p에서 출발하여 교차형 구조적 커널 B로서 필요한 만큼의 팽창을 한다(그림 10.6에서 사용한 것과 같이). 이를 계속 반복하기 전에, 각각의 팽창 연산 후에 $\bar{A}$와 교집합을 취하면서 팽창 연을 반복한다. 이렇게 하여 아래와 같은 집합의 수열을 만든다.

$$\{p\} = X_0, X_1, X_2, \ . \ . \ . \ , X_k = X_{k+1},$$

이 과정에서 X_n은 아래와 같이 계산된다.

$$X_n = (X_{n-1} \oplus B) \cap \overline{A}.$$

최종적으로 $X_k \cup A$은 채워진 영역이다. 그림 10.22는 이를 보여준다. 그림 10.22 (b)에서 아래와 같이 영역을 채워 나간다.

$$X_0 = \{p\}, \quad X_1 = \{p, 1\}, \quad X_2 = \{p, 1, 2\}, \ . \ . \ .$$

교차형 구조의 커널을 이용하는 것은 대각형 경계는 제외하는 것을 의미

안겹치는 부분이 새번호를 붙이고, 새번호에 필터를 올리고,..., ∞ B=

	1	
1	1	1
	1	

A

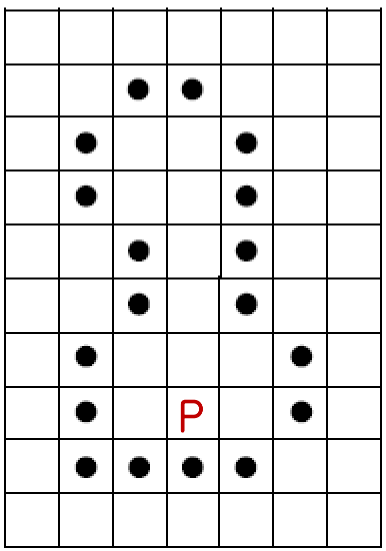


FIGURE 10.21 An 8-connected boundary of a region to be filled.

$\overline{A}$:

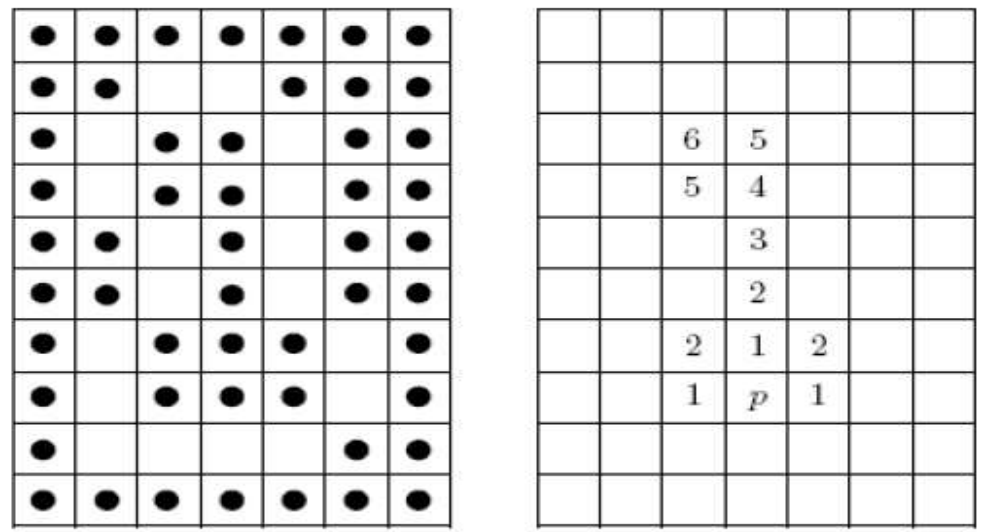


그림 10.22 영역의 채우기 과정

10.6.2 연결 성분

연결된 성분을 채우기 위해 매우 유사한 알고리즘을 사용한다. 4-연결 성분을 위해 교차형 구조의 커널을 이용하고, 8-연결 성분을 위해 정방형 구조의 커널을 사용한다. 화소 p에서 시작하면, 집합의 수열을 만들면서 해당 성분의 나머지를 아래와 같이 채운다.

$$X_0 = \{p\}, X_1, X_2, \ldots,$$

이때 Xn은 아래와 같이 계산되며, Xk = Xk−1이 될 때까지 계산된다.

$$X_n = (X_{n-1} \oplus B) \cap A$$

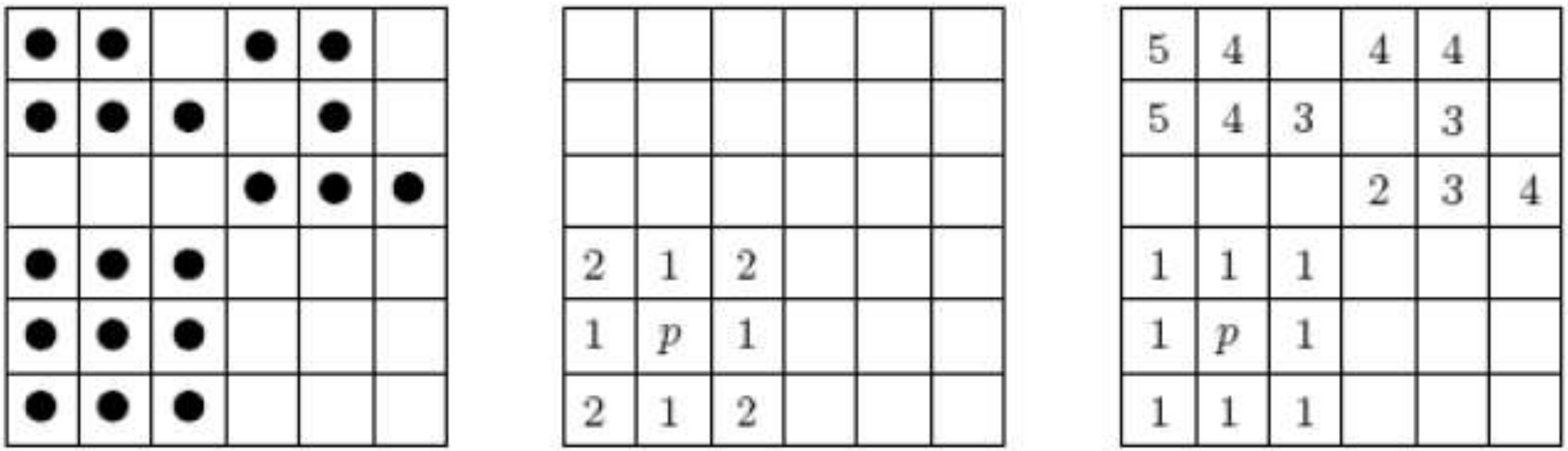
그림 10.23은 이 예를 나타낸다. 각 경우에 왼쪽 아래에서 정방형의 중심에서 시작한다. 왜냐하면 정방형은 자신이 4-연결 성분이고, 교차형 커널은 이를 건널 수 없기 때문이다.

4연결

	1	
1	1	1
	1	

8연결

1	1	1
1	1	1
1	1	1



Using the cross　　Using the square

그림 10.23 연결 성분 채우기

이들 2개의 알고리즘은 MATLAB 함수로 쉽게 구현될 수 있다. 영역 채우기를 구현하기 위해, 아래의 list와 같이 2개 영상 current 및 previous의 경로를 유지하고, 이들 사이에 차분이 없으면 정지한다. 해당 영역에서 단일 점 p를 previous로 하고, current를 팽창 연산 $(p \oplus B) \cap \bar{A}$로 하여 시작한다. 다음 단계로 아래와 같이 셋팅한다.

$$\texttt{previous} \leftarrow \texttt{current},$$

$$\texttt{current} \leftarrow (\texttt{current} \oplus B) \cap \overline{A}.$$

B가 주어지면 MATLAB으로 마지막 단계를 아래와 같이 구현한다.

```
imdilate(current,B)&~A.
```

이 함수는 그림 10.24와 같다.

```
function out=regfill(im,pos,kernel)
% REGFILL(IM,POS,KERNEL) performs region filling of binary
% image IMAGE,with kernel KERNEL, starting at point with
% coordinates given by POS.
% Example:
%            n=imread('nicework.tif');
%            nb=n&~imerode(n,ones(3,3));
%            nr=regfill(nb,[74,52],ones(3,3));
%
current=zeros(size(im));
last=zeros(size(im));
last(pos(1),pos(2))=1;
current=imdilate(last,kernel)&~im;
while any(current(:)~=last(:)),
  last=current;
  current=imdilate(last,kernel)&~im;
end;
out=current;
```

그림 10.24 영역 채우기를 위한 간단한 프로그램 list

우리는 경계를 따라 특정한 영역의 윤곽을 채우기 위해 이것을 아래와 같이 사용할 수 있다.

```
>> n=imread('nicework.tif');
>> imshow(n),pixval on
>> nb=n&~imerode(n,sq);
>> figure,imshow(nb)
>> nf=regfill(nb,[74,52],sq);
>> figure,imshow(nf)
```

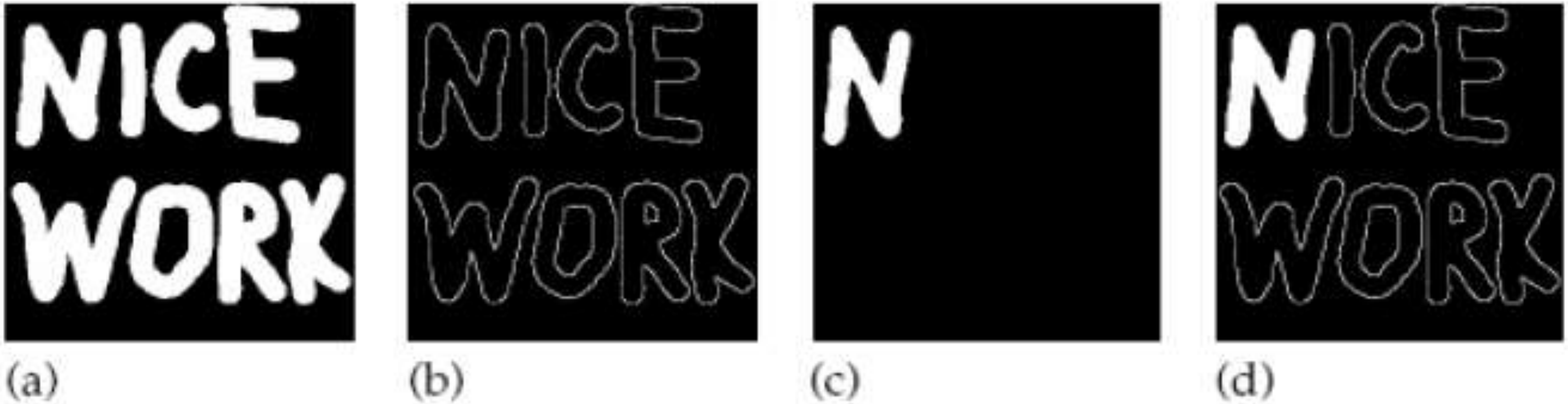


그림 10.25 영역 채우기 과정

10.6.3 골격화 처리(Skeletonization)

이진 영상에서 물체의 골격은 그 물체의 사이즈와 모양을 캡슐로 보호하는 라인과 곡선들의 모임이다. 하나의 주어진 물체에 대하여 골격을 정의하는 방법은 사실상 여러 가지 방법이 있는데 많은 다른 골격이 존재한다. 우리는 제 10장에서 몇 가지 살펴볼 것이다. 그러나 골격은 형태학적 방법을 이용하여 아주 간단하게 얻을 수 있다. 아래의 표 10.1과 같이 연산 표를 생각하자.

여기서 동일한 구조적 커널 B를 사용하는 k배의 침식 연산을 $A \ominus kB$로 사용하는 것이 좋다. $(A \ominus kB) \circ B$가 공집합이 될 때까지 표를 계속 연산한다. 그 후에 골격은 모든 집합의 차분들의 합집합을 취하여 얻는다. 예로서 교차형 구조 커널을 이용한 그림 10.28을 얻을 수 있다.

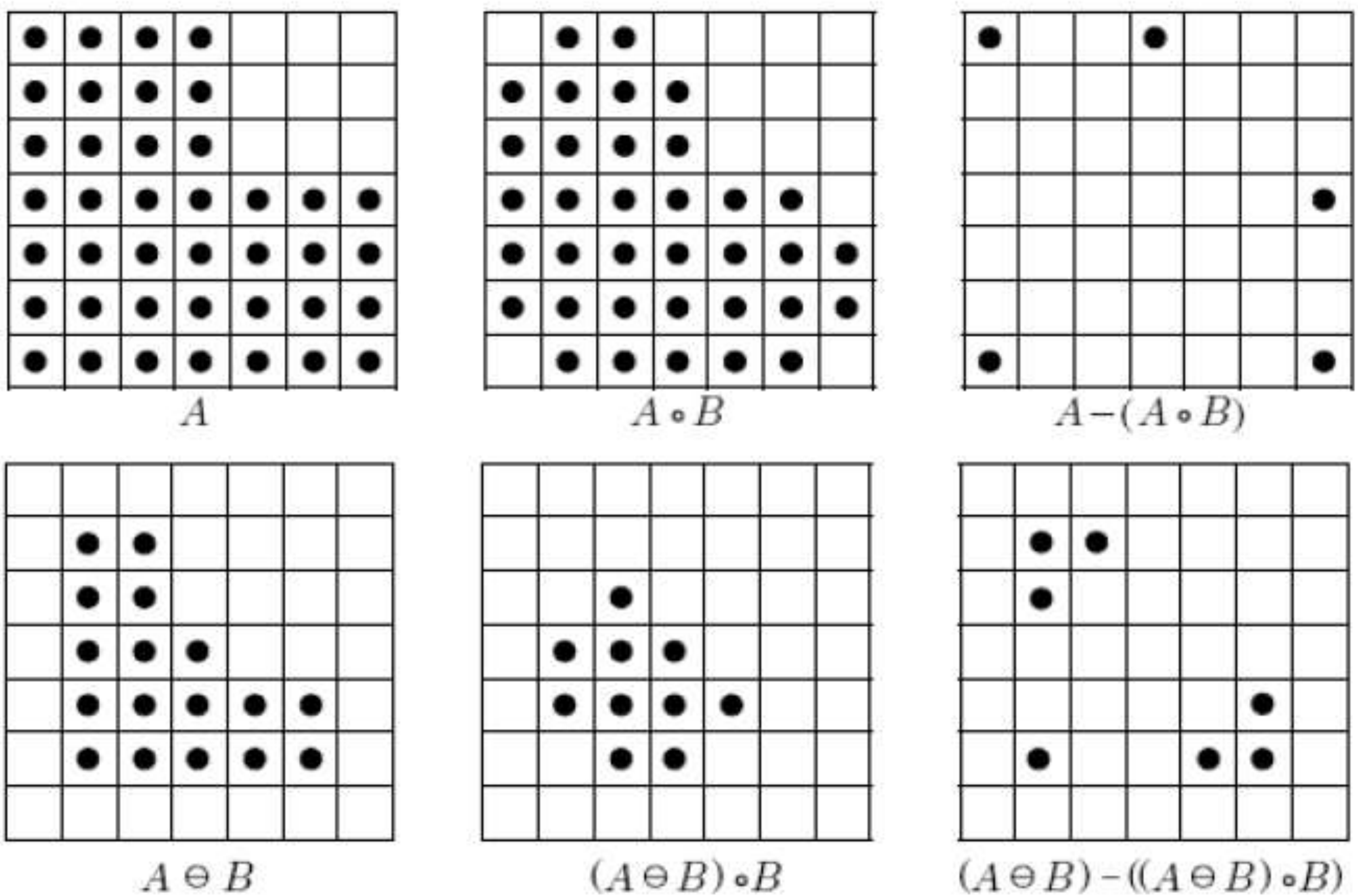
$(A \ominus 2B) \circ B$가 공집합이므로 여기서 정지한다. 골격은 3번째 열에서 모든 집합들의 합집합이다. 이것을 그림 10.29에 나타내었다. 이 골격화의 방법을 Lantuéjuol's method라 한다.

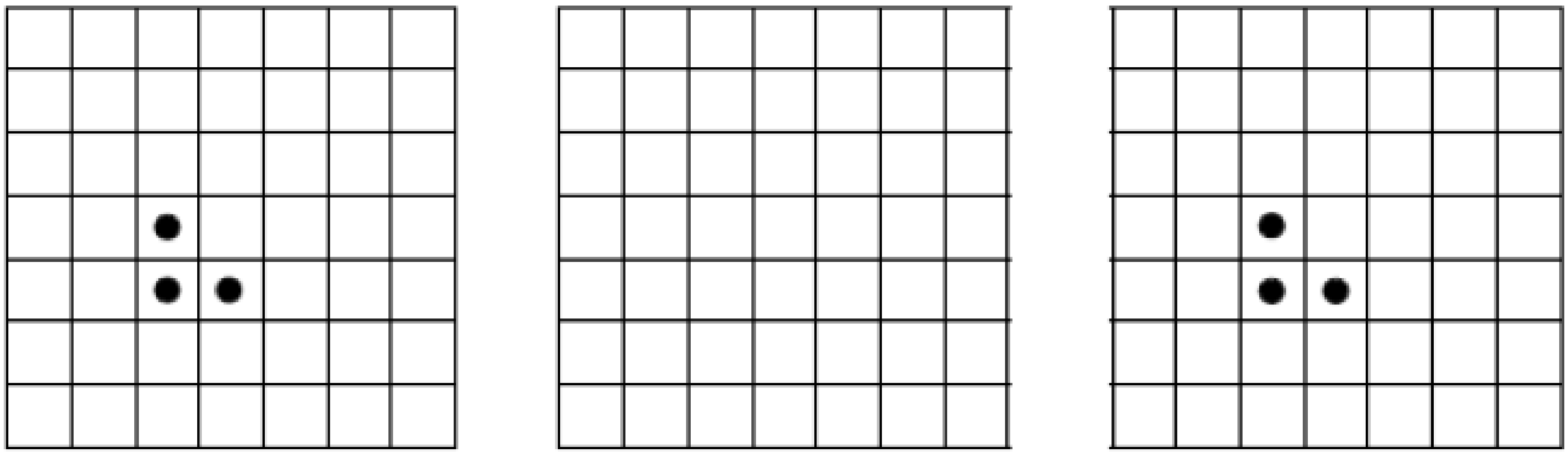
⊖ 침식
∘ open

TABLE 10.1 *Operations used to construct the skeleton.*

Erosions	Openings	Set differences
A	$A \circ B$	$A - (A \circ B)$
$A \ominus B$	$(A \ominus B) \circ B$	$(A \ominus B) - ((A \ominus B) \circ B)$
$A \ominus 2B$	$(A \ominus 2B) \circ B$	$(A \ominus 2B) - ((A \ominus 2B) \circ B)$
$A \ominus 3B$	$(A \ominus 3B) \circ B$	$(A \ominus 3B) - ((A \ominus 3B) \circ B)$
$\vdots$	$\vdots$	$\vdots$
$A \ominus kB$	$(A \ominus kB) \circ B$	$(A \ominus kB) - ((A \ominus kB) \circ B)$

공집합이 될 때까지 하고 ( ) 값을 합집합


A
$A \circ B$
$A-(A \circ B)$
$A \ominus B$
$(A \ominus B) \circ B$
$(A \ominus B)-((A \ominus B) \circ B)$



$A \ominus 2B$　　$(A \ominus 2B) \circ B$　　$(A \ominus 2B) - ((A \ominus 2B) \circ B)$

그림 10.28 골격화 처리 과정

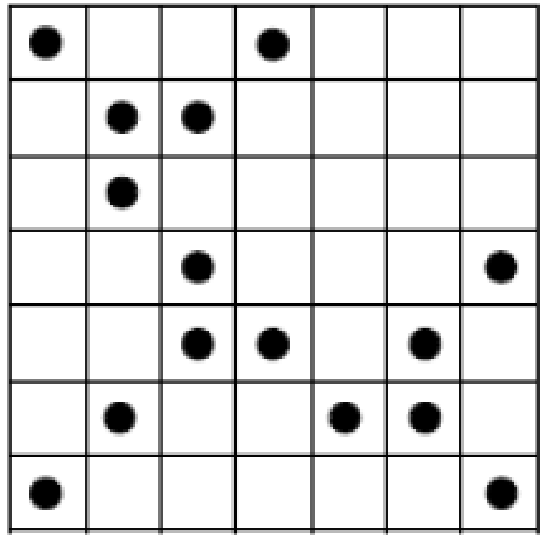


그림 10.29 골격화 처리 결과

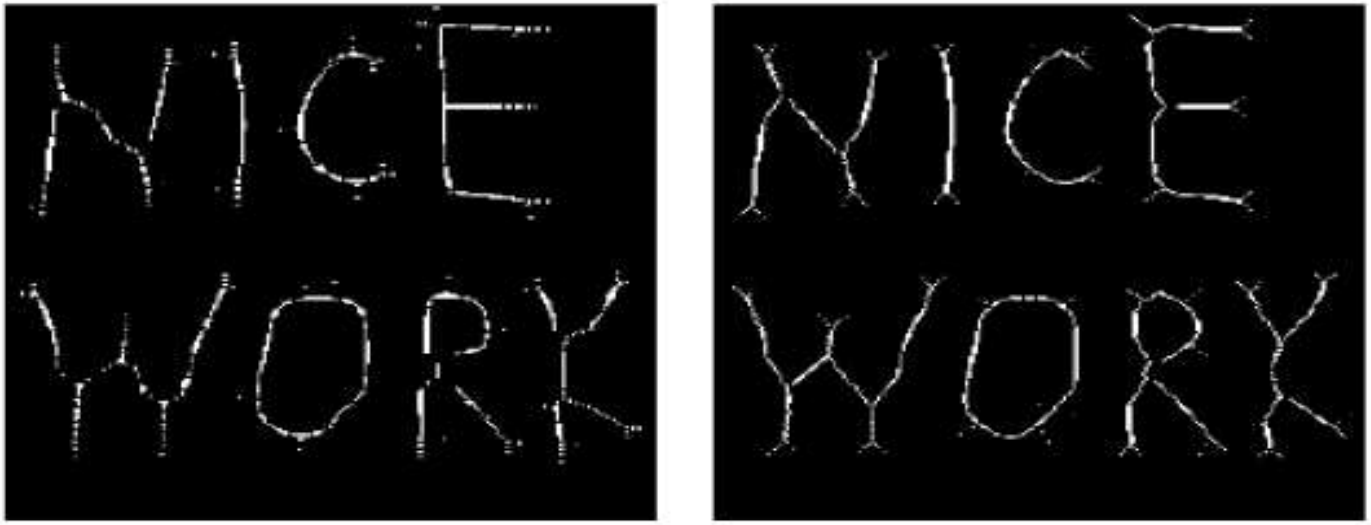
```
function skel = imskel(image,str)
% IMSKEL(IMAGE,STR) - Calculates the skeleton of binary image IMAGE using
% structuring element STR.  This function uses Lantejoul's algorithm.
%
skel=zeros(size(image));
e=image;
while (any(e(:))),
    o=imopen(e,str);
    skel=skel | (e&~o);
    e=imerode(e,str);
end
```

그림 10.30 골격화 계산의 간단한 프로그램 list

이 프로그램은 그림 10.30과 같이 함수를 이용하면 매우 쉽게 구현될 수 있다. "nice work" 영상을 실험한 것이다.

```
>> nk=imskel(n,sq);
>> imshow(nk)
>> nk2=imskel(n,cr);
>> figure,imshow(nk2)
```

이 결과는 그림 10.31에 보였다. 그림 (a)는 정방형 구조의 커널을, (b)는 교차형 구조의 커널을 이용한 결과이다.



(a) (b)

그림 10.31 이진 영상의 골격화 처리 결과