# Morphology

김성영교수
금오공과대학교
컴퓨터공학과

# 학습 목표

- Morphology의 의미를 설명할 수 있다.
- Dilation과 Erosion 연산의 용도와 동작을 설명할 수 있다.
- Opening과 Closing 연산의 용도와 동작을 설명할 수 있다.

# 모폴로지|Morphology 개요

- 모폴로지 (형태학)
  - 생물학의 한 분야로 동물이나 식물의 모양이나 구조를 다루는 학문

- 수학적 모폴로지|mathematical morphology
  - 관심 객체의 검출을 쉽게 처리할 수 있도록 영상 분할 결과를 단순화하는 방법으로 사용
  - 객체 경계의 단순화, 작은 구멍을 채움, 작은 돌기의 제거 등
  - Binary 영상과 Gray-scale 영상에 적용 가능
  - 모폴로지 필터링morphological filtering
    - 구조적 요소structuring element와 팽창dilation 및 침식erosion 연산 사용

# 기본 집합 이론Basic Set Theory

Let $A$ and $B$ be sets in $Z^2$

$a$ is an **element** of $A$ ➡ $a \in A$

$a$ is **not an element** of $A$ ➡ $a \notin A$

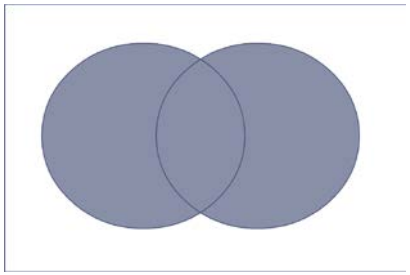
$A$ is a **subset** of $B$ ➡ $A \subseteq B$

The **union** of $A$ and $B$ ➡ $A \cup B = \{x \mid x \in A \text{ or } x \in B\}$

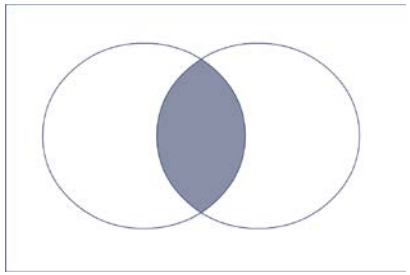
The **intersection** of $A$ and $B$ ➡ $A \cap B = \{x \mid x \in A \text{ and } x \in B\}$

The **complement** of $A$ ➡ $A^c = \{x \mid x \notin A\}$

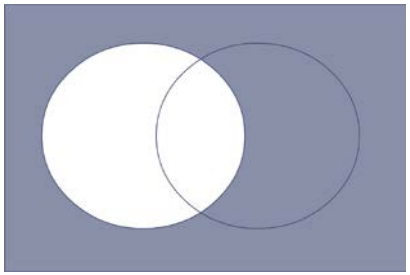
The **difference** of $A$ and $B$ ➡ $A - B = \{x \mid x \in A \text{ and } x \notin B\}$



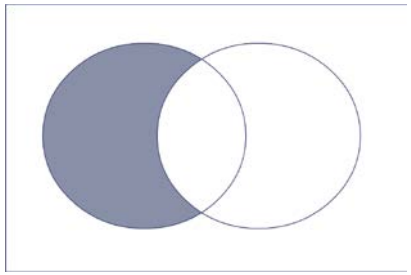
$A \cup B$



$A \cap B$



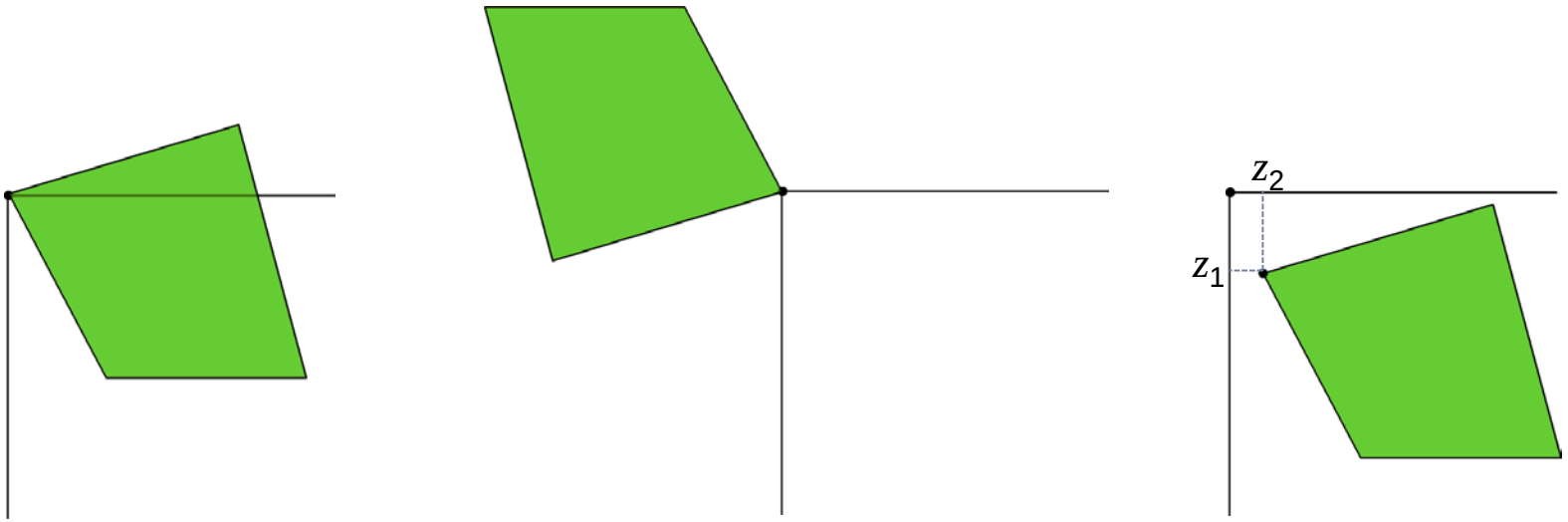
$A^c$



$A$-$B$

The **reflection** of $A$ ➡ $\hat{A} = \{x \mid x = -b, \text{ for } b \in A\}$

The **translation** of $A$ ➡ $(A)_z = \{c \mid c = b + z, \text{ for } b \in A\}$




reflection translation

# 이진 영상에서의 팽창 연산Dilation operation

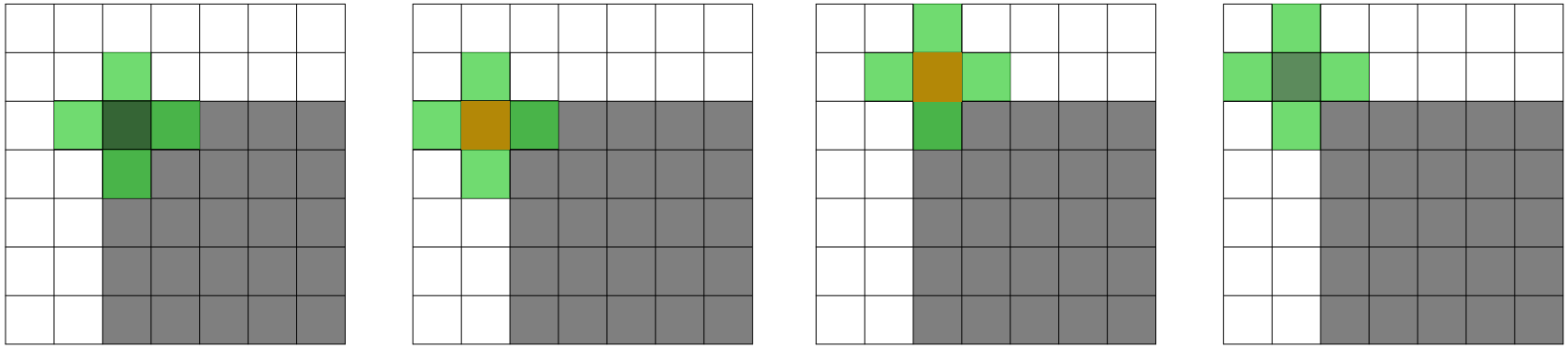
- 객체의 크기를 확장
  - 객체 내부의 작은 구멍을 채움
  - 근접한 위치의 두 객체를 연결

$$A \oplus B = \left\{ z \mid \left(\hat{B}\right)_z \cap A \neq \emptyset \right\}$$

$\boldsymbol{A}$: image
$\boldsymbol{B}$: Structuring element

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

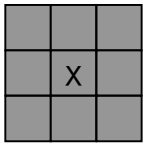
$$A \oplus (B \oplus C) = (A \oplus B) \oplus C$$



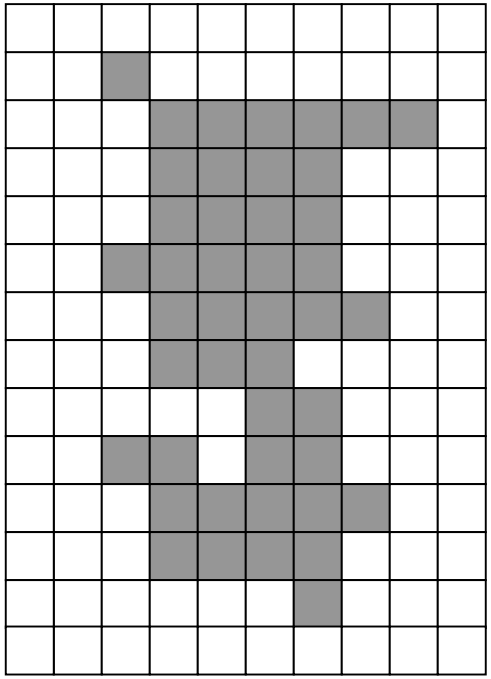
# 알고리즘

1. 구조적 요소의 중심이 영상의 ‘0’에 위치하면 다음 위치로 이동
2. 구조적 요소의 중심이 영상의 ‘1’에 위치하면 구조요소와 영상을 논리적 OR 연산 수행

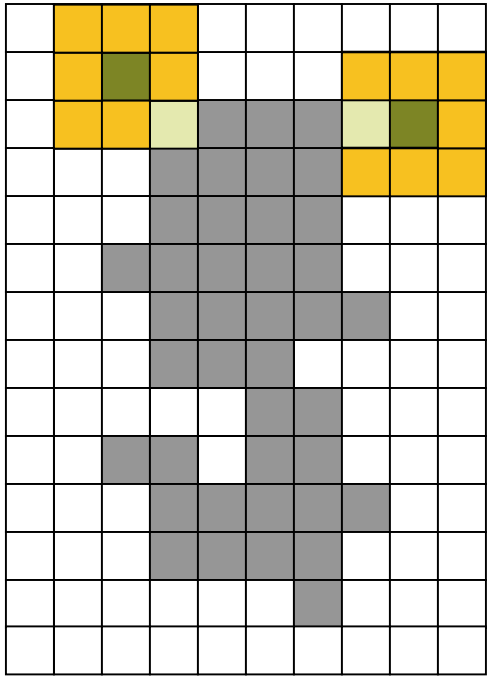
# example



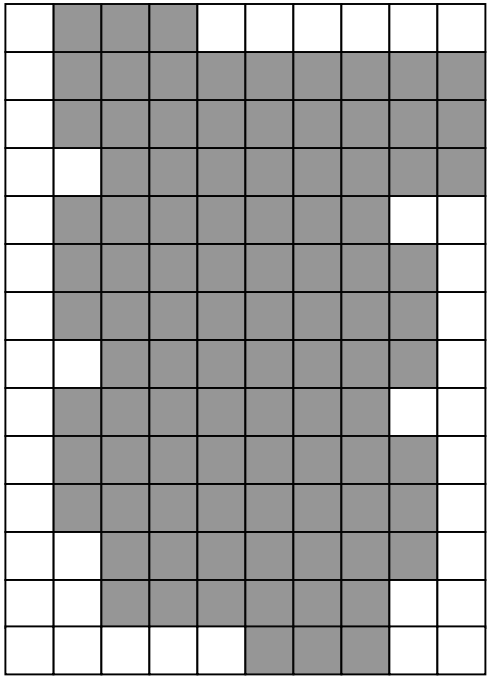

Structuring Element



Original Image

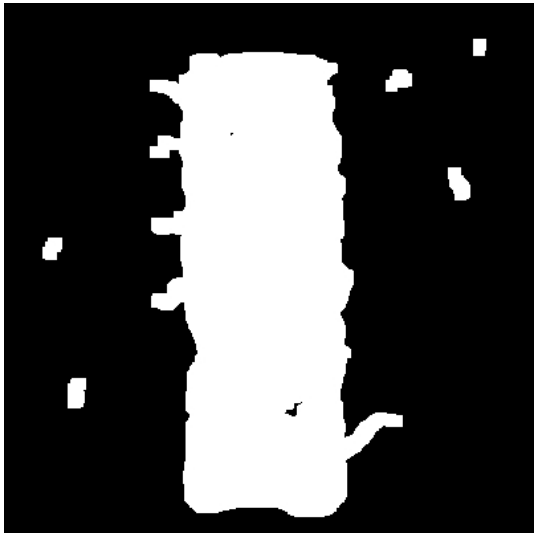


중간 결과



After Dilation

Original image



Dilation with SE
of rectangle (7x7)

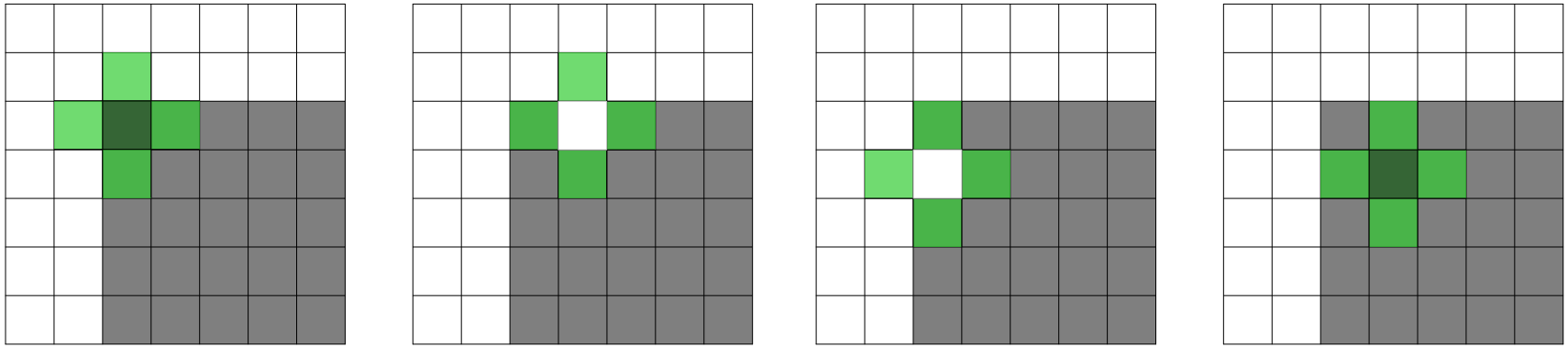
Dilation with SE
of circle (7x7)

# 이진 영상에서의 침식 연산Erosion operation

- 객체의 크기를 축소
  - 객체 경계를 침식
  - 작은 돌기를 제거

$$A \ominus B = \{z \mid B_z \subseteq A\}$$

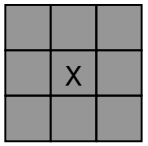
$\boldsymbol{A}$: image
$\boldsymbol{B}$: Structuring element



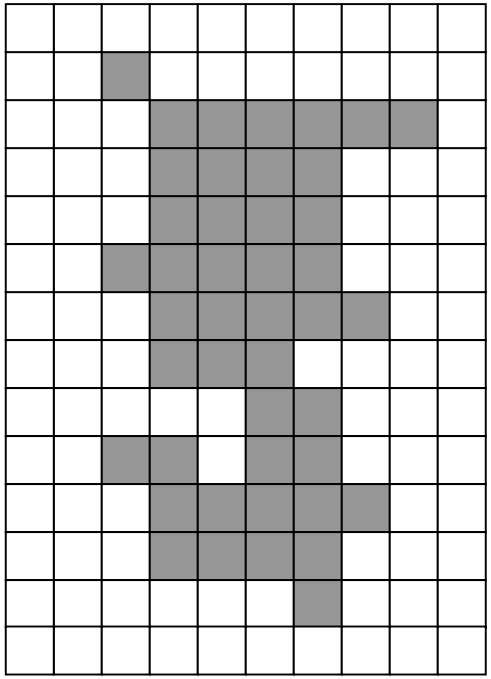
## 알고리즘

1. 구조적 요소의 중심이 영상의 ‘0’에 위치하면 다음 위치로 이동
2. 구조적 요소의 중심이 영상의 ‘1’에 위치하면 구조요소에서 ‘1’ 위치가 하나라도 객체를 벗어나면 그 위치는 ‘0’으로 변경

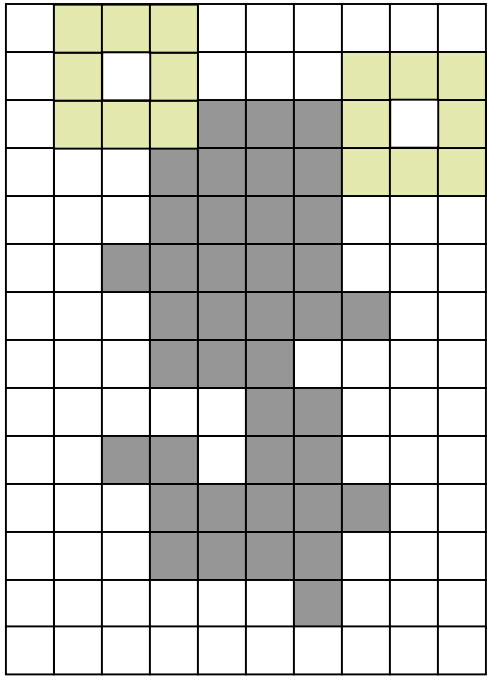
# example



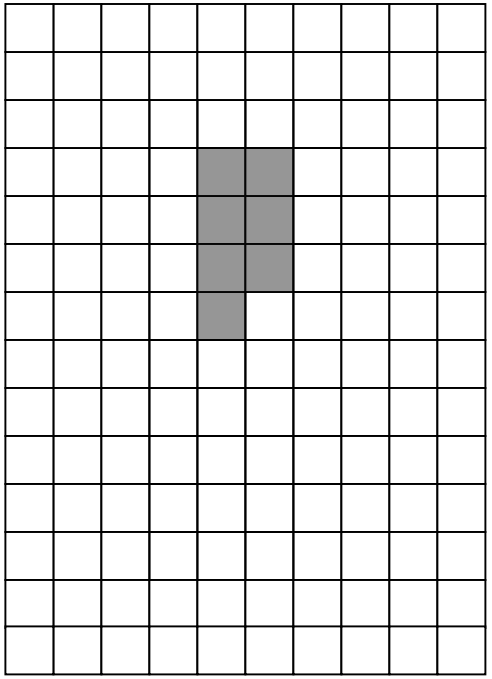

Structuring Element



Original Image

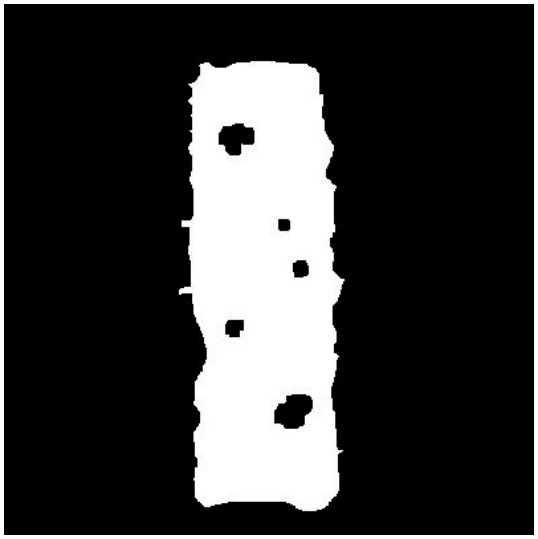


중간 결과

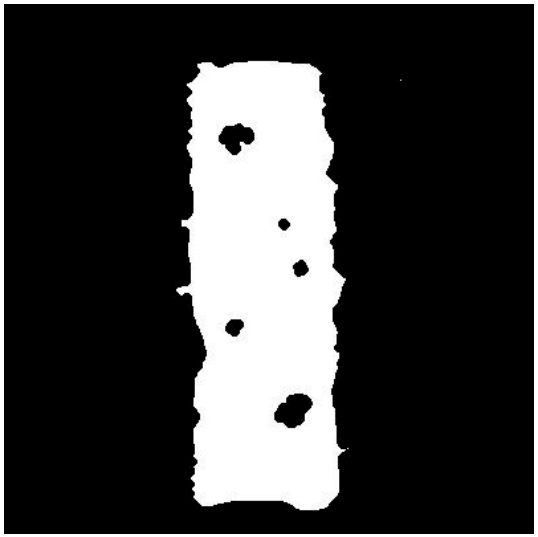


After Erosion

Original image



Erosion with SE
of rectangle (7x7)



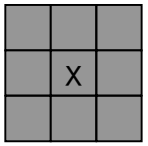
Erosion with SE
of circle (7x7)

# 열림 연산Opening operation

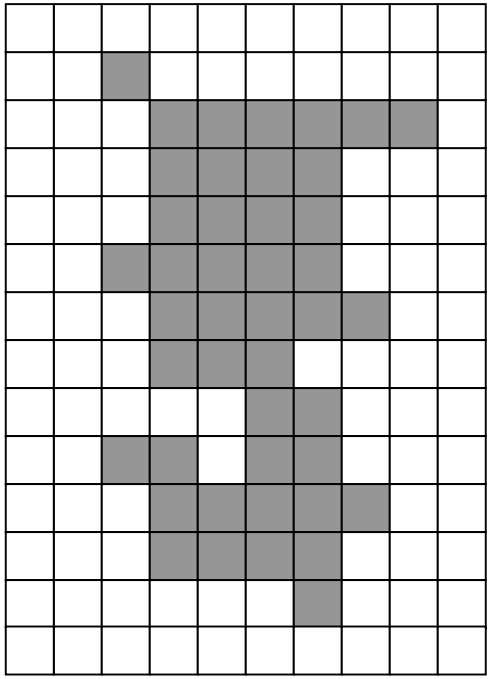
- 침식Erosion 연산을 수행한 후 다시 팽창Dilation 연산 적용
- 작은 크기의 객체에 포함되는 픽셀들을 제거

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

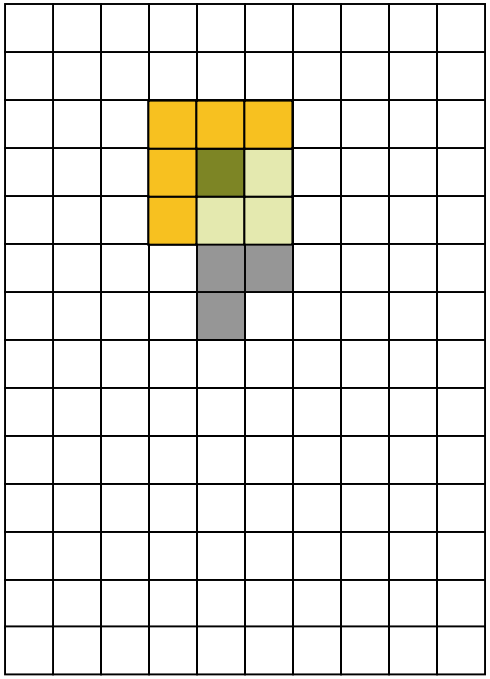
# example



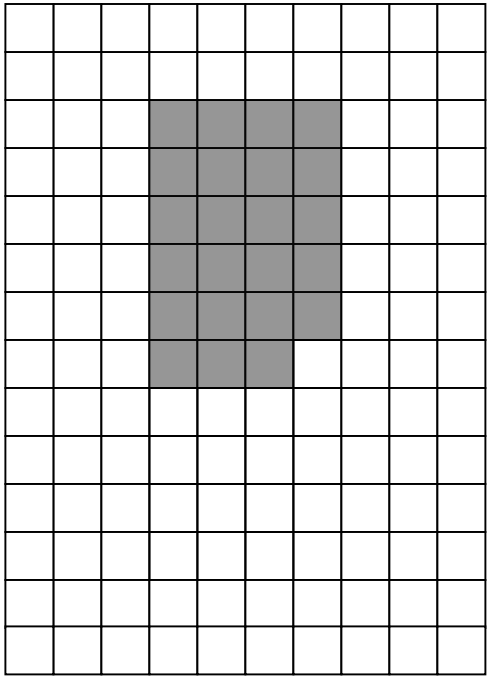

Structuring Element



Original Image



After Erosion



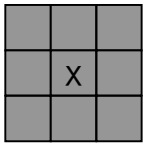
After Opening

# 닫힘 연산Closing operation

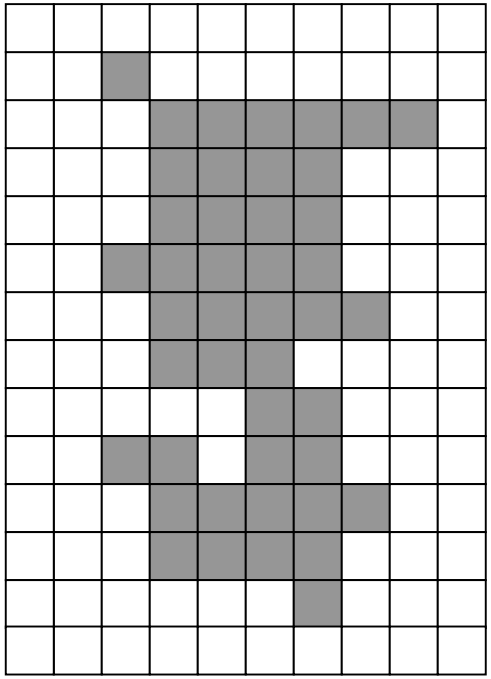
- 팽창Dilation 연산을 수행한 후 다시 침식Erosion 연산 적용
- 객체 내부의 작은 구멍hole이나 간격gap을 채움

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

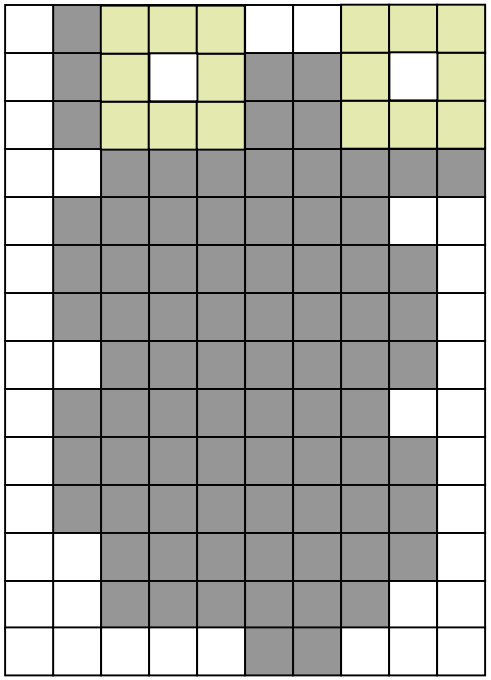
# example



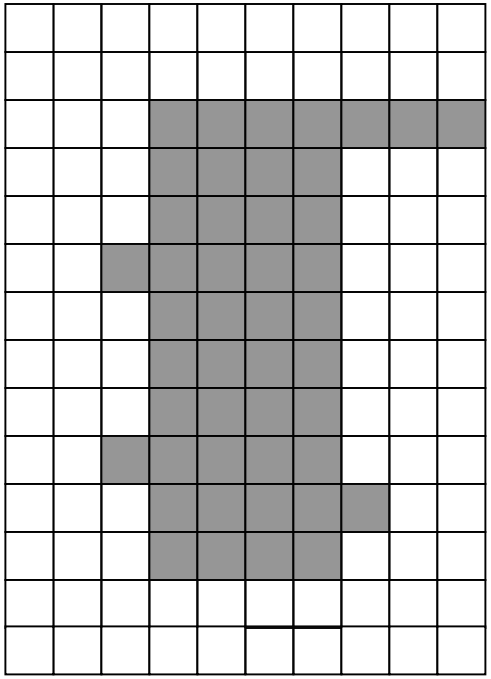

Structuring Element



Original Image

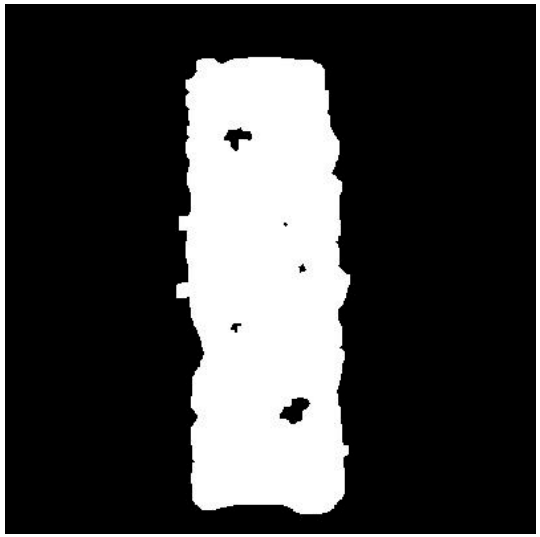


After Dilation



After Closing

Original image

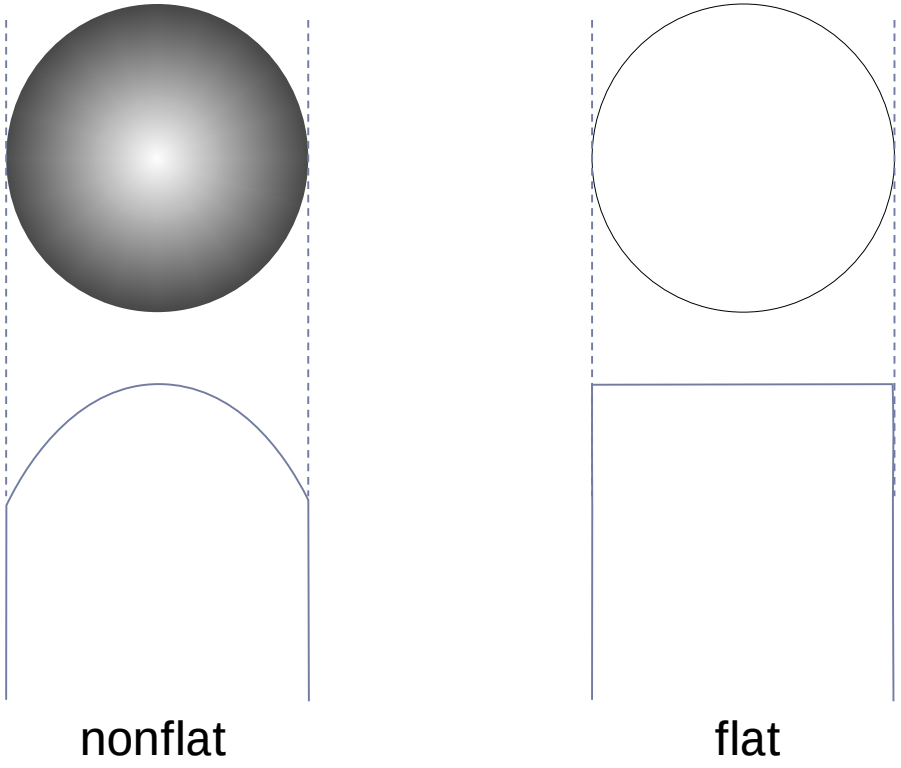


Opening with SE
of rectangle (7x7)

Closing with SE
of rectangle (7x7)

# Operations for gray-scale images

Two types of a structuring element



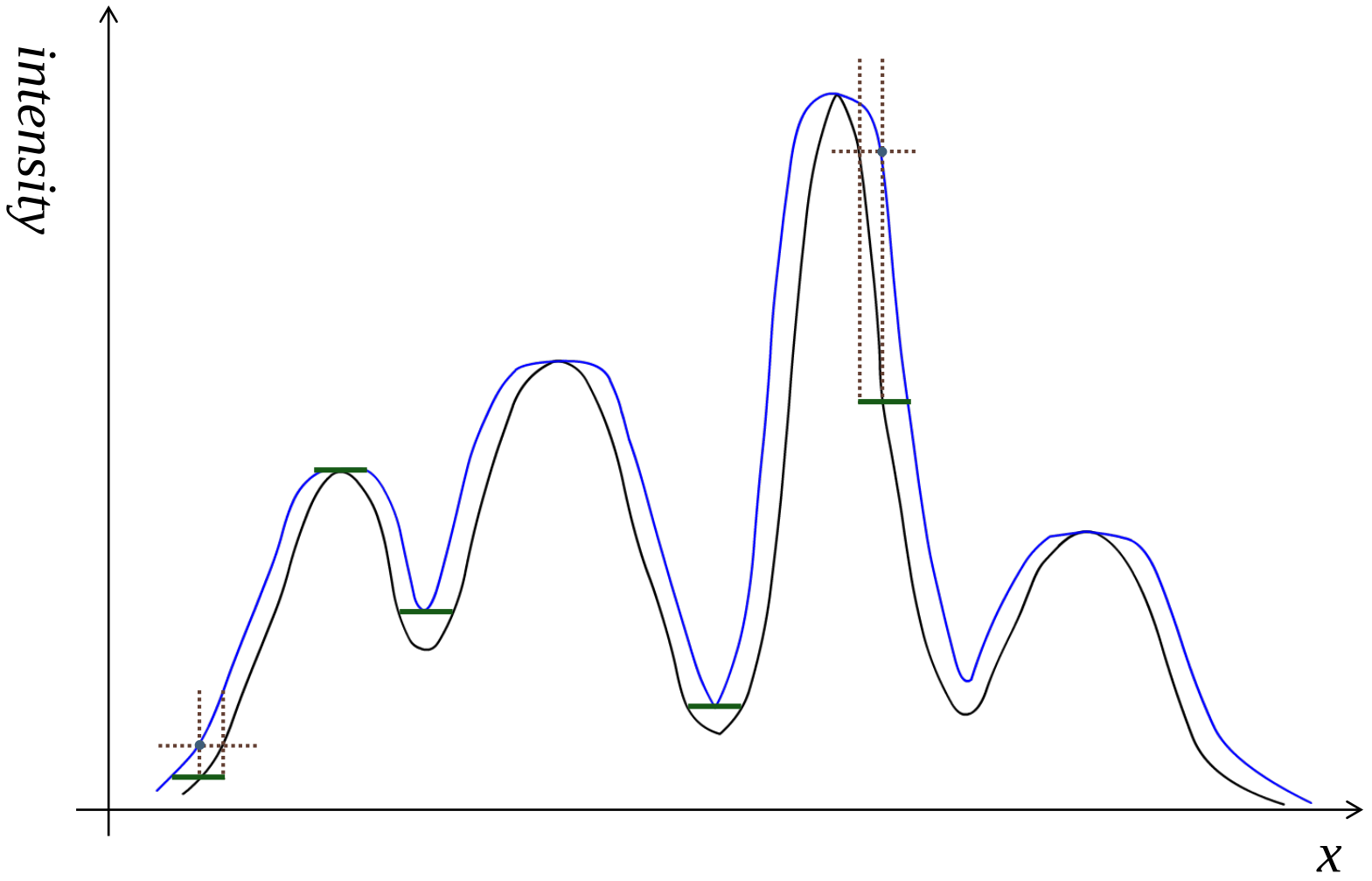
**Dilation** of $f$ by $k$

$$(f \oplus k)(s,t) = \max_{(s,t)\in D_k} \{f(x-s, y-t) + k(s,t)\}$$

**Erosion** of $f$ by $k$

$$(f \ominus k)(s,t) = \min_{(s,t)\in D_k} \{f(x+s, y+t) - k(s,t)\}$$

# Dilation with a flat structuring element



# OpenCV를 사용한 Morphological operations 구현

```
Mat   getStructuringElement( … )
void  dialte(  …  )
void  erode(  …  )
void  morphologyEx(  …  )
```

```
Mat getStructuringElement(
      int shape,
      Size ksize,
      Point anchor=Point(-1,-1)
)
```

shape

- MORPH_RECT:  a rectangular structuring element
- MORPH_ELLIPSE: an elliptic structuring element,

  Rect( 0, 0, ksize.width, ksize.height )
- MORPH_CROSS: a cross-shaped structuring element
- CV_SHAPE_CUSTOM: custom structuring element

```
void dilate(
      InputArray src,
      OutputArray dst,
      InputArray kernel,
      Point anchor=Point(-1,-1),
      int iterations=1,
      int borderType=BORDER_CONSTANT,
      const Scalar& borderValue =
                        morphologyDefaultBorderValue()
)
```

```
void erode(
      InputArray src,
      OutputArray dst,
      InputArray kernel,
      Point anchor=Point(-1,-1),
      int iterations=1,
      int borderType=BORDER_CONSTANT,
      const Scalar& borderValue =
                        morphologyDefaultBorderValue()
)
```

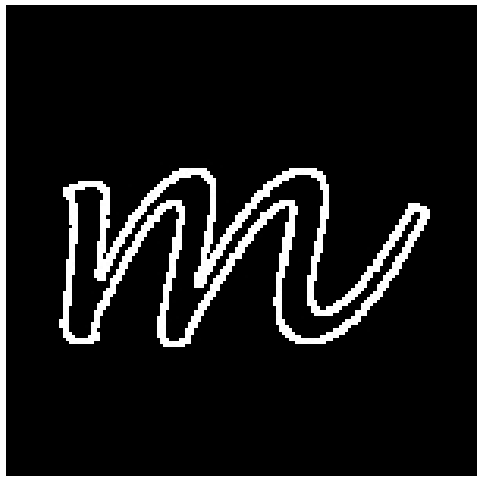
```
void morphologyEx(
      InputArray src,
      OutputArray dst,
      int op,
      InputArray kernel,
      Point anchor=Point(-1,-1),
      int iterations=1,
      int borderType=BORDER_CONSTANT,
      const Scalar& borderValue =
                  morphologyDefaultBorderValue()
)
```

op

- MORPH_OPEN - an opening operation
- MORPH_CLOSE - a closing operation
- MORPH_GRADIENT - a morphological gradient
  $( f \oplus b ) - ( f \ominus b )$
- MORPH_TOPHAT - “top hat” $f - (f \circ b)$
- MORPH_BLACKHAT - “black hat” $( f \bullet b ) - f$

dilation

erosion



gradient

dilation

erosion

```
void Dilation( const Mat &image, Mat &result, int type=0, int size=3 );
void Erosion( const Mat &image, Mat &result, int type=0, int size=3 );
void Morphology( const Mat &image, Mat &result, int op, int type=0, int size=3 );

int main(void){
     Mat image = imread( "text_m.bmp", -1 );
     if( image.data == NULL ) return -1;

     Mat dilation;
     Dilation( image, dilation );

     Mat erosion;
     Erosion( image, erosion );

     Mat morphology;
     Morphology( image, morphology, 2 );

     // Display the images
     namedWindow( "Image" );
     namedWindow( "Dilation" );
     namedWindow( "Erosion" );
     namedWindow( "Morphology" );
     imshow( "Image", image );
     imshow( "Dilation", dilation );
     imshow( "Erosion", erosion );
     imshow( "Morphology", morphology );

     waitKey();

     return 0;
}
```

```
void Dilation( const Mat &image, Mat &result, int type, int size )
{
        // allocate if necessary
       result.create( image.size(), image.type() );

       int dilation_type;
       if( type == 0 )
              dilation_type = MORPH_RECT;
       else if( type == 1 )
              dilation_type = MORPH_CROSS;
       else if( type == 2)
              dilation_type = MORPH_ELLIPSE;

       Mat element = getStructuringElement(
                        dilation_type, Size(size, size) );

       // Apply the dilation operation
       dilate( image, result, element );
}
```

```
void Erosion( const Mat &image, Mat &result, int type, int size )
{
        // allocate if necessary
       result.create( image.size(), image.type() );

       int erosion_type;
       if( type == 0 )
                erosion_type = MORPH_RECT;
       else if( type == 1 )
                erosion_type = MORPH_CROSS;
       else if( type == 2)
                erosion_type = MORPH_ELLIPSE;

       Mat element = getStructuringElement(
                        erosion_type, Size(size, size) );

       // Apply the dilation operation
       erode( image, result, element );
}
```

```
void Morphology( const Mat &image, Mat &result, int op,
 int type, int size ) {
     // allocate if necessary
    result.create( image.size(), image.type() );

    int operation;
    switch( op )
    {
    case 0:
        operation = MORPH_OPEN;
        break;
    case 1:
        operation = MORPH_CLOSE;
        break;
    case 2:
        operation = MORPH_GRADIENT;
        break;
    case 3:
        operation = MORPH_TOPHAT;
        break;
    case 4:
        operation = MORPH_BLACKHAT;
        break;
    }
```

```
        int mor_type;
        if( type == 0 )
                mor_type = MORPH_RECT;
        else if( type == 1 )
                mor_type = MORPH_CROSS;
        else if( type == 2)
                mor_type = MORPH_ELLIPSE;

        Mat element = getStructuringElement( mor_type,
                Size(size, size) );

        // Apply the specified morphology operation
        morphologyEx( image, result, operation, element );
}
```

# Reference


- R. Gonzalez, R. Woods, **Digital Image Processing (2nd Edition),** Prentice Hall, 2002
- Scott E Umbaugh, **Computer Imaging**, CRC Press, 2005
- R. Laganière, **OpenCV2 Computer Vision: Application Programming Cookbook,** PACKT Publishing, 2011
- http://docs.opencv.org