暑期課程
基本影像處理 day6-1

指導教授：顏淑惠、林慧珍
http://163.13.127.10
http://pria.cs.tku.edu.tw

指導教授：凃瀞珽
http://mail.tku.edu.tw/cttu

2015.07.08

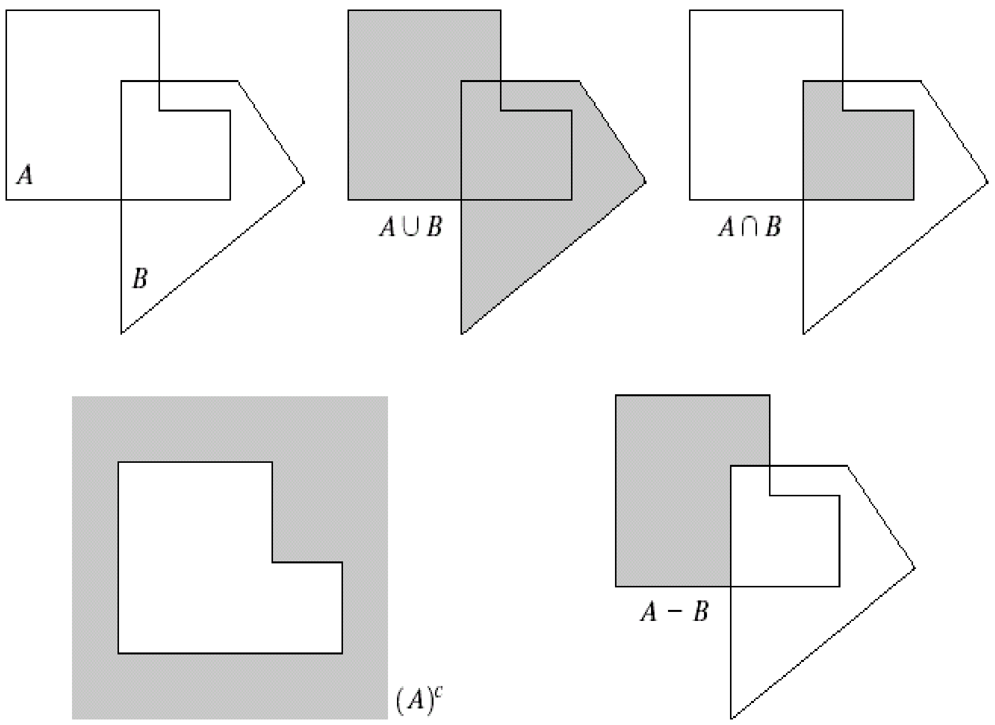
Course Outline

- 型態影像學
  - Dilation (膨脹)
  - Erosion (侵蝕)
  - Opening and Closing

- Connected Component Labeling
  - (連通元件標記法)

Image Morphology

- 型態影像學，又稱型態學影像處理
  - 針對binary image中的像素執行集合運算(set operation)



a b c
d e

FIGURE 9.1 (a) Two sets A and B. (b) The union of A and B. (c) The intersection of A and B. (d) The complement of A. (e) The difference between A and B.

Image Morphology

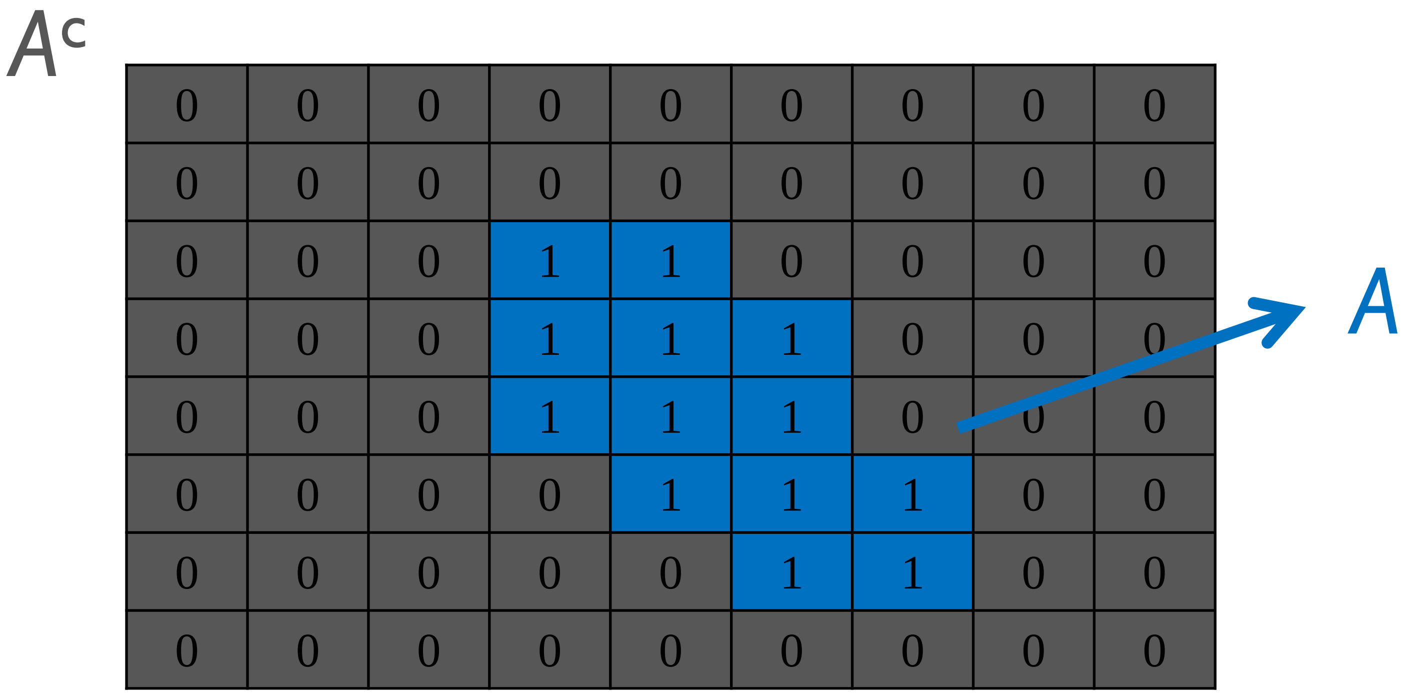
- 在離散空間(discrete space) Z^2，考慮一張二值化影像為一個具有兩個離散變數(x, y)的函數。我們定義影像中的物件A 為所有具有共同屬性的像素的集合:
  $\downarrow$ *I(x, y)*
  - $A = \{ a \mid property(a) == TRUE \}$

TRUE = 1;
FALSE = 0;

- 至於A的背景則表示為A^c (A的補集) :
  - $A^c = \{ a \mid a \notin A \}$

a: a pixel point at (x, y) in image.
property(a): pixel value in *binary image* at point a

$$A = \{\, \mathrm{a} \mid \mathit{property}(\mathrm{a}) == 1 \,\}$$

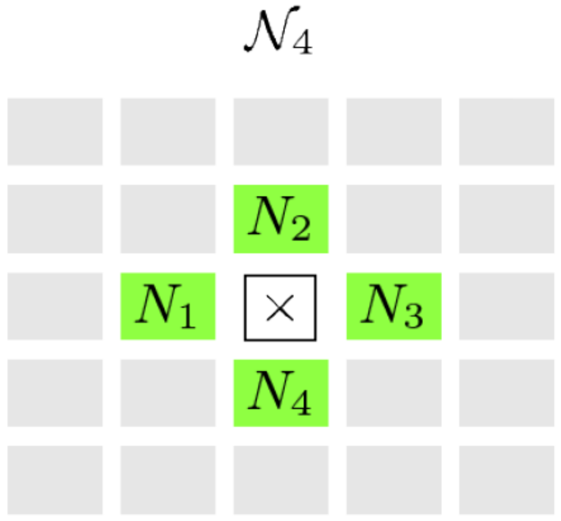


0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0
0	0	0	1	1	1	0	0	0
0	0	0	1	1	1	0	0	0
0	0	0	0	1	1	1	0	0
0	0	0	0	0	1	1	0	0
0	0	0	0	0	0	0	0	0

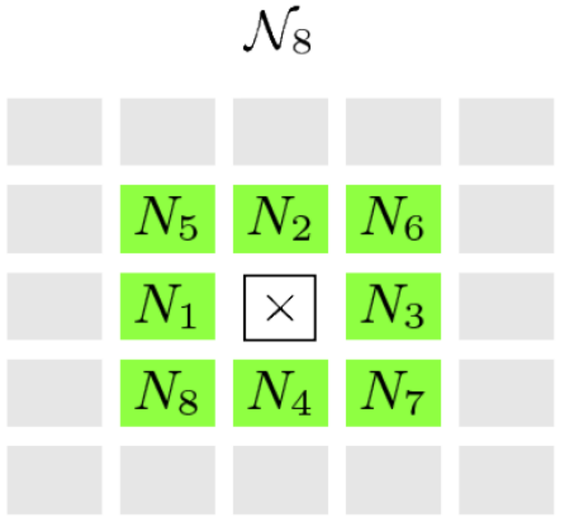
```
TRUE = 1;          //foreground, object, target, etc.
FALSE = 0;         //background
```

Pixel Neighborhoods

- the two definitions of "neighbors"



4 Neighborhood



8 Neighborhood

Dilation & Erosion

- 型態學影像處理常用的基本運算
  - 膨脹(dilation)和侵蝕(erosion)。

- 其運算的原理是將影像A與一個區域視窗(local window)—稱為結構元素B (structuring element) 的特定集合運算，其結果為一個新的濾波影像。

常見結構元素：

	1	
1	1	1
	1	

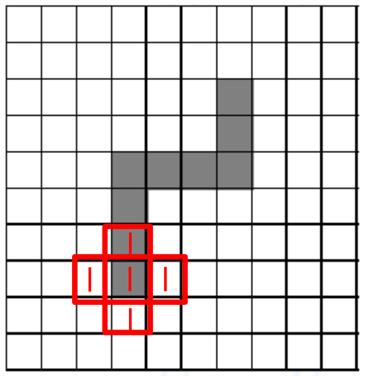
1	1	1
1	1	1
1	1	1

Dilation

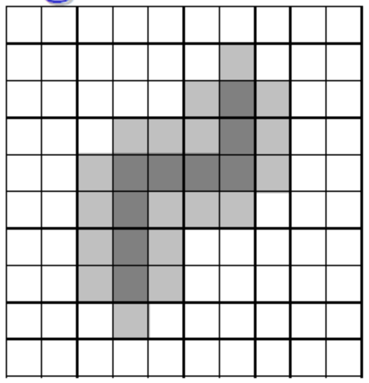
- 膨脹(dilation)

$$D(A,B) = A \oplus B = \bigcup_{b \in B} (A+b)$$

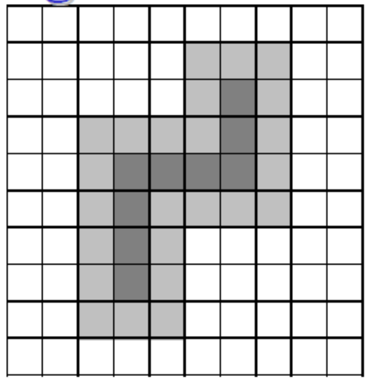
Original image



4-neighbor Dilation



8-neighbor Dilation



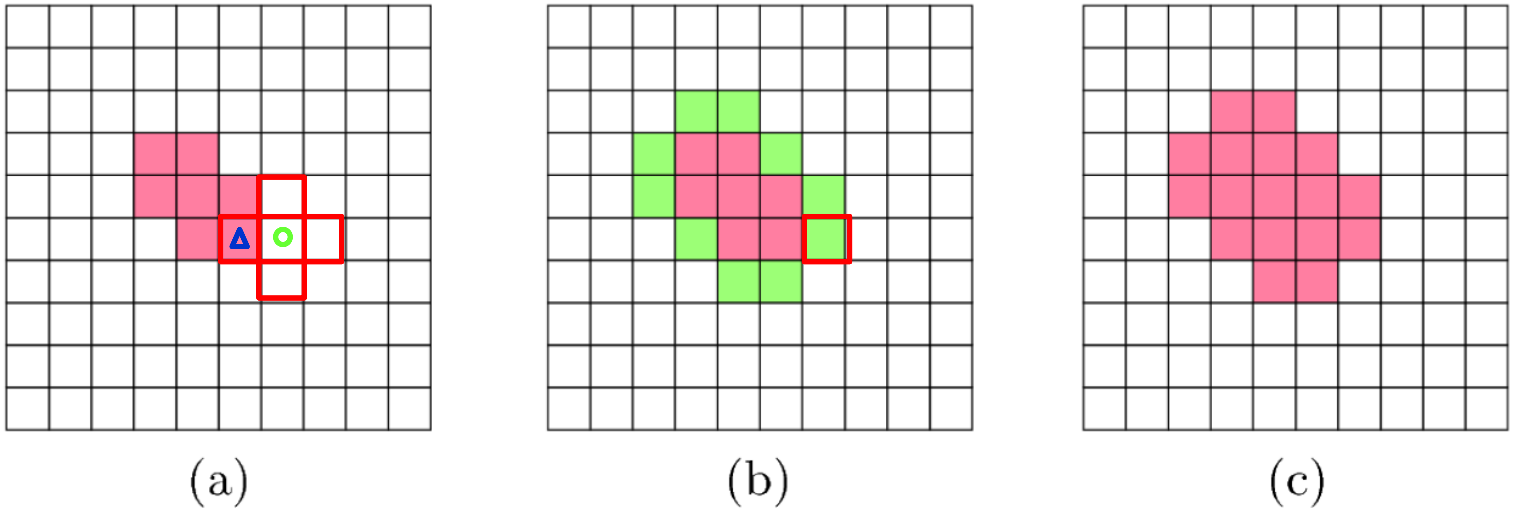
	1	
1	1	1
	1	

1	1	1
1	1	1
1	1	1

Dilation

- 方法：
  - 如果某個背景像素的4-neighbor或8-neighbor中有任何一個像素之素值為1，則將該像素之像素值以1取代，亦即將此像素加入至原始圖形內。
  - 也就是說，若中間元素為1(前景像素)則 4-neighbor或8-neighbor也會為1。

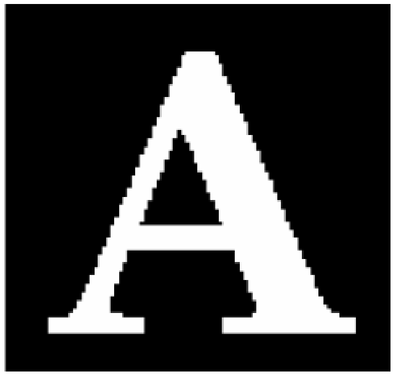
Dilation

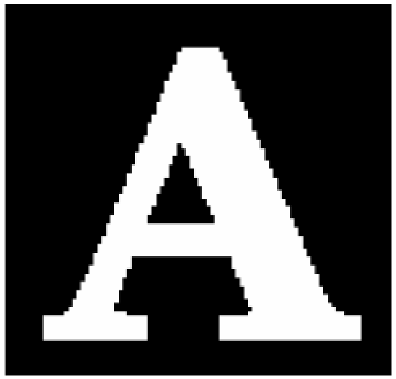


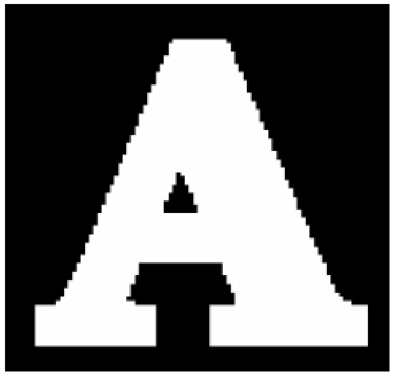
Change a background pixel to foreground if it has a foreground pixel as a 4-neighbor.

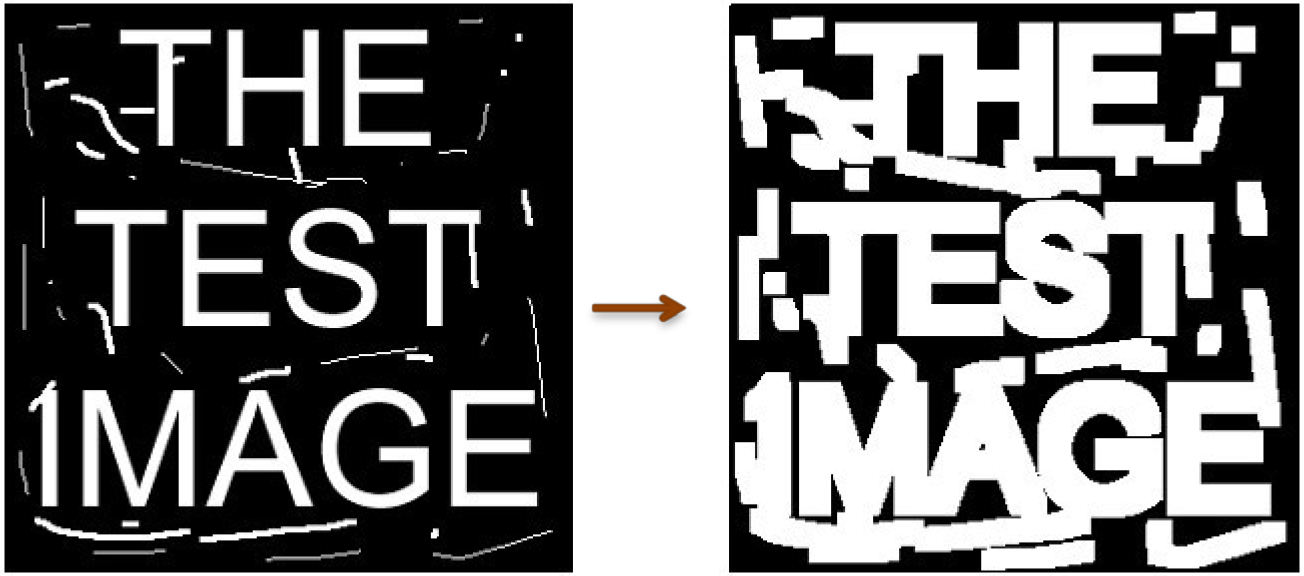
Dilation

PS:白色部分灰階值為1


original


Iteration once


Iteration 3 times

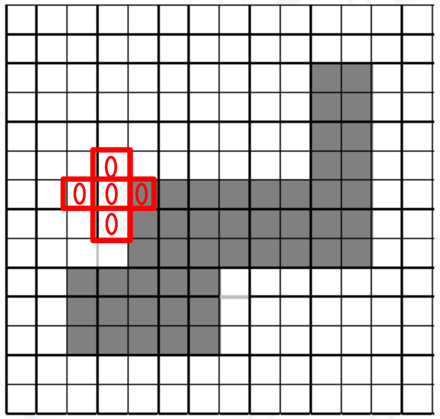


Erosion

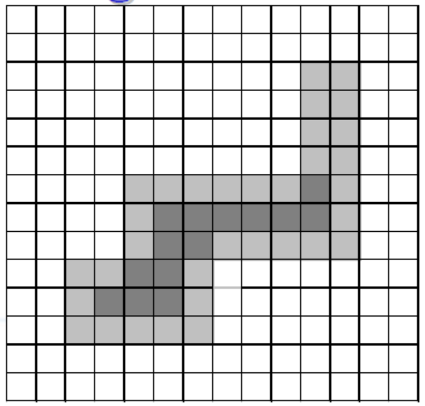
- 侵蝕(erosion)

$$E(A,B) = A\Theta(-B) = \bigcap_{b\in B}(A-b)$$

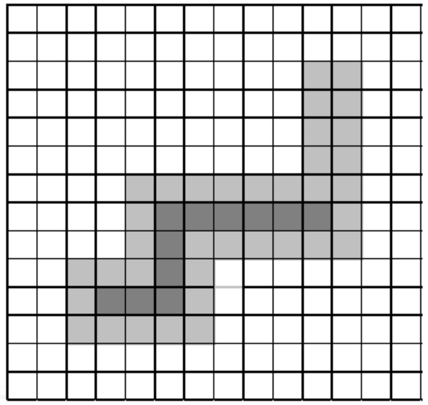
Original image



4-neighbor Erosion



8-neighbor Erosion



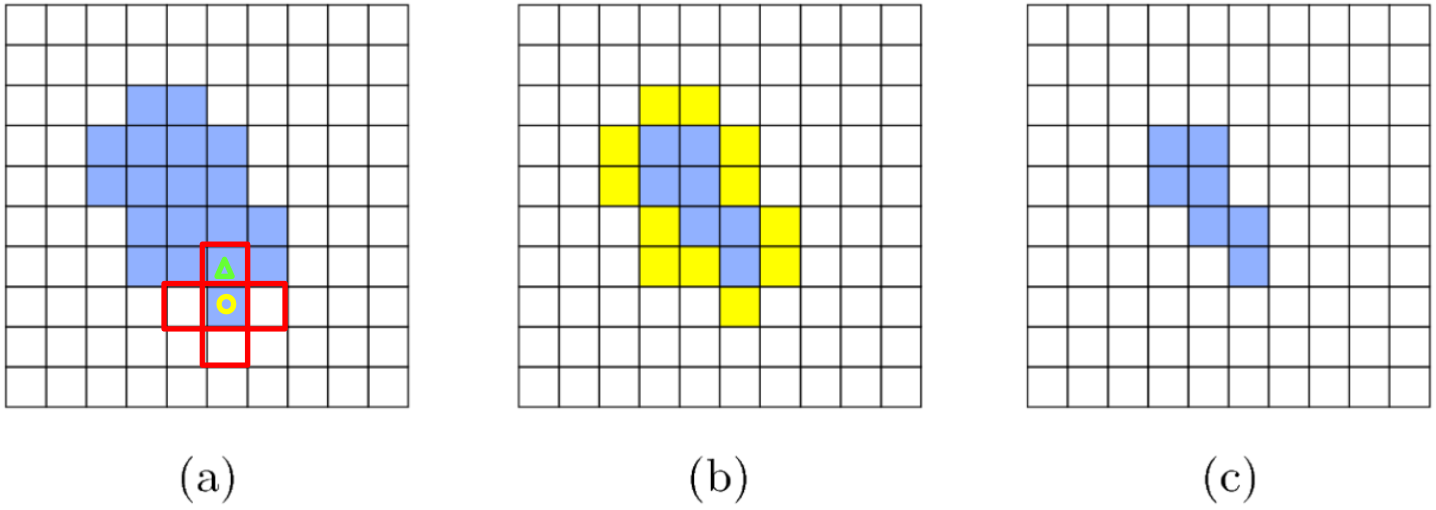
	0	
0	0	0
	0	

0	0	0
0	0	0
0	0	0

Erosion

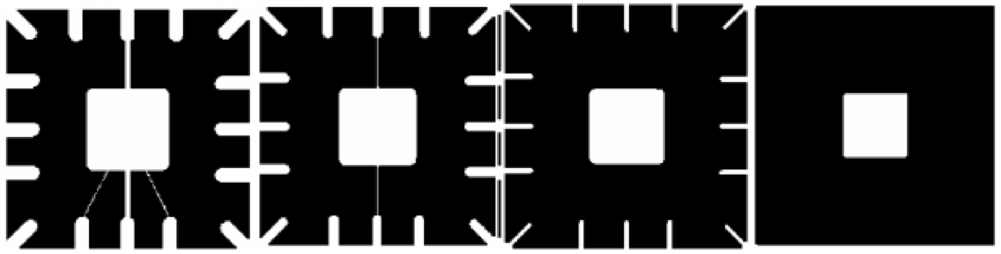
- 方法：
  - 如果某個前景像素的4-neighbor或8-neighbor中有任何一個像素之素值為0，則將該像素之像素值以0取代，亦即將此像素從原始圖形內去掉。
  - 也就是說，若4-neighbor或8-neighbor有任一元素為0 (背景像素)，則中間元素為0。

Erosion

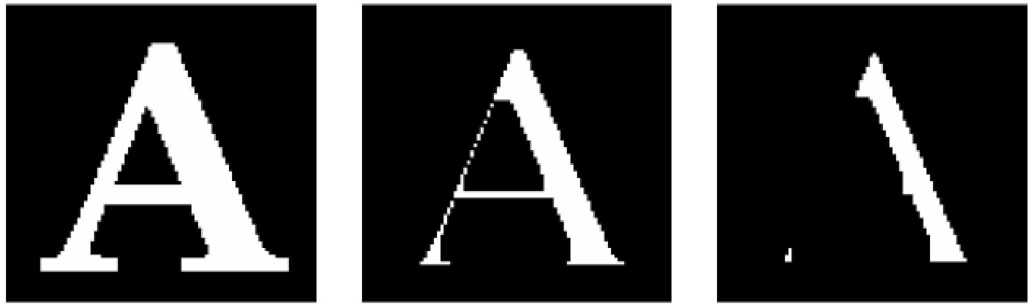


Change a foreground pixel to background if it has a background pixel as a 4-neighbor.

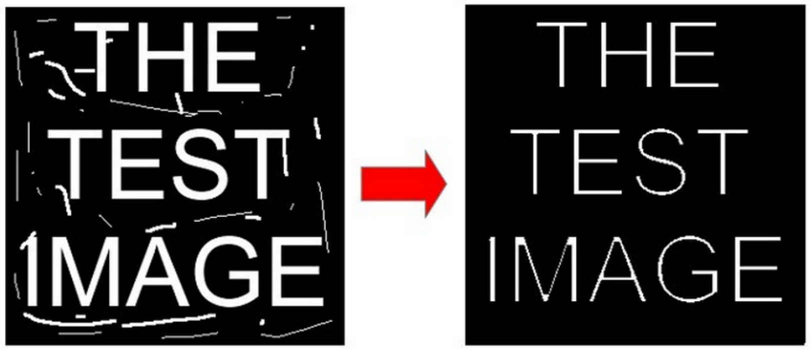
Erosion



Original R_B=5 R_B=10 R_B=20



original Iteration 2 times



Course Outline

- 型態影像學
  - Dilation (膨脹)
  - Erosion (侵蝕)
  - Opening and Closing
- A short introduction to Pattern Recognition
- Histogram of Oriented Gradients

Opening and Closing

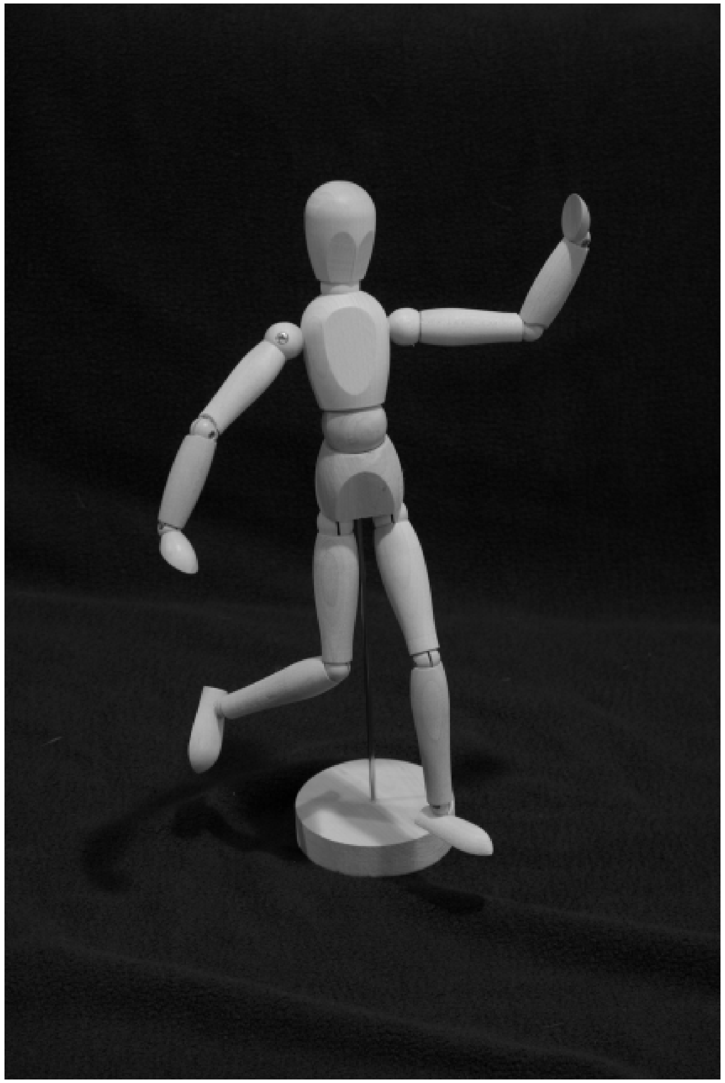
- 建立在dilation和erosion運算的基礎上，可以建構出更高階的型態學運算：
  - 斷開(Opening)

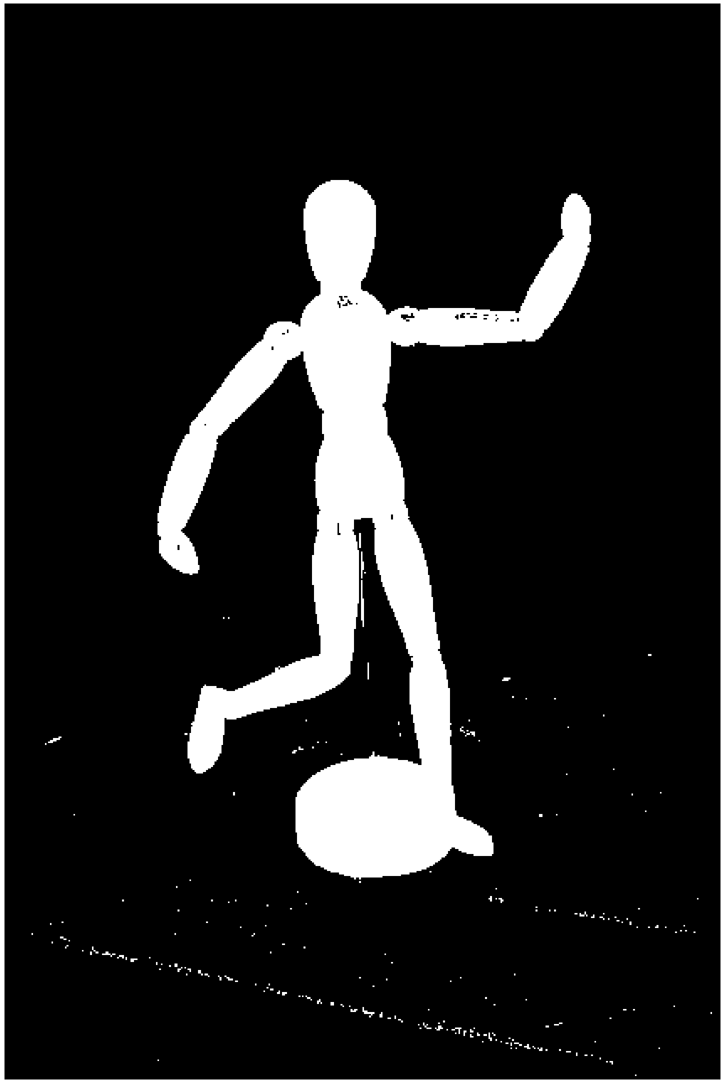
$$I \circ H = (I \ominus H) \oplus H$$

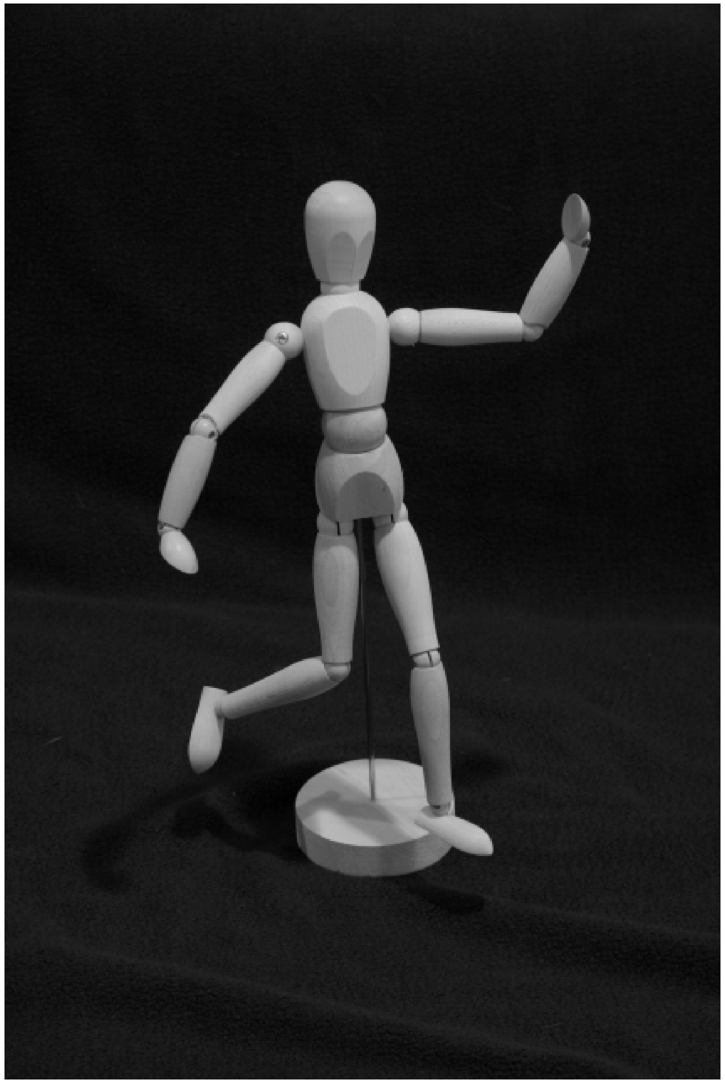
  - 閉合(Closing)

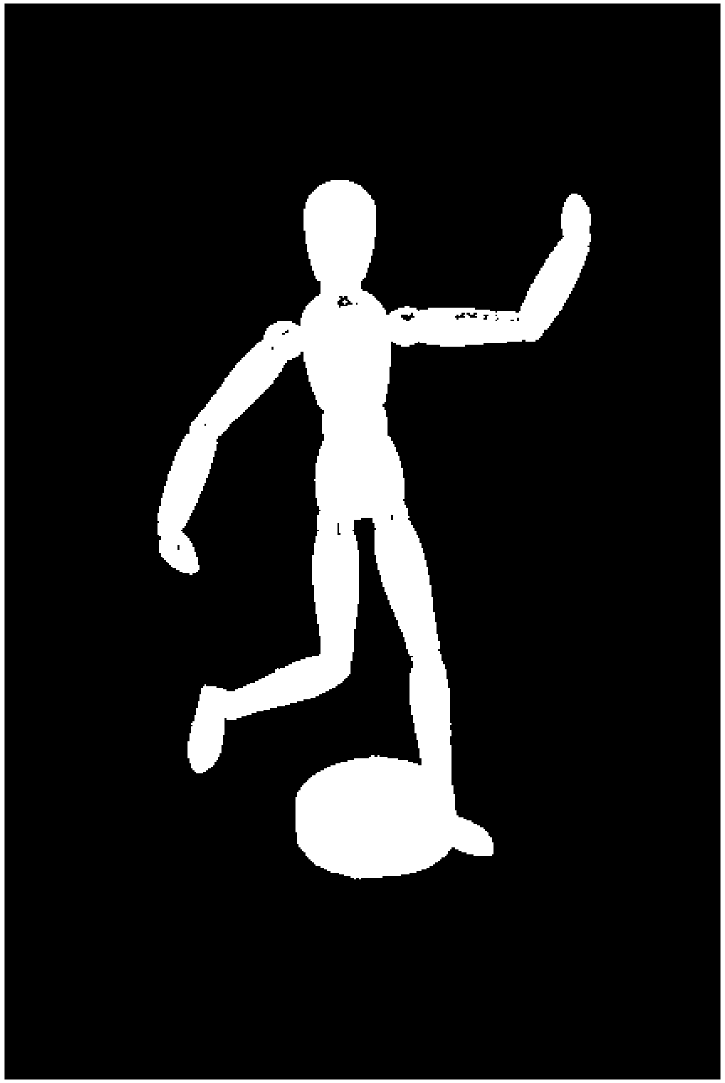
$$I \bullet H = (I \oplus H) \ominus H$$

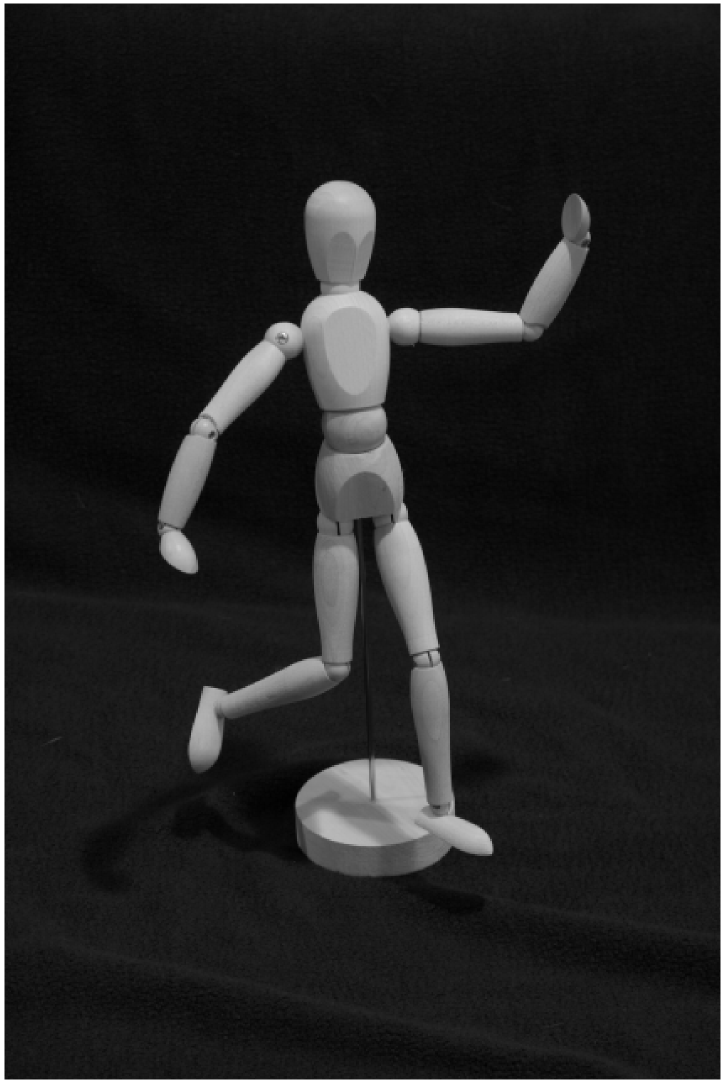
  - *I*: image
  - *H*: structure element


Original image

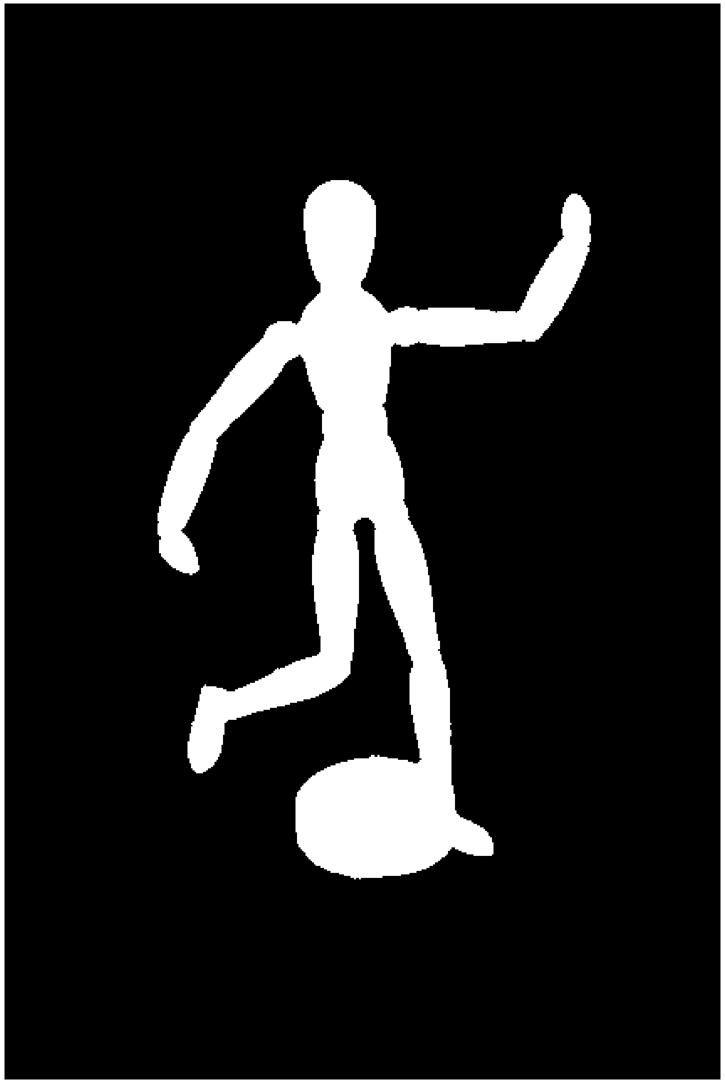

Initial threshold


Original image


After opening



Original image

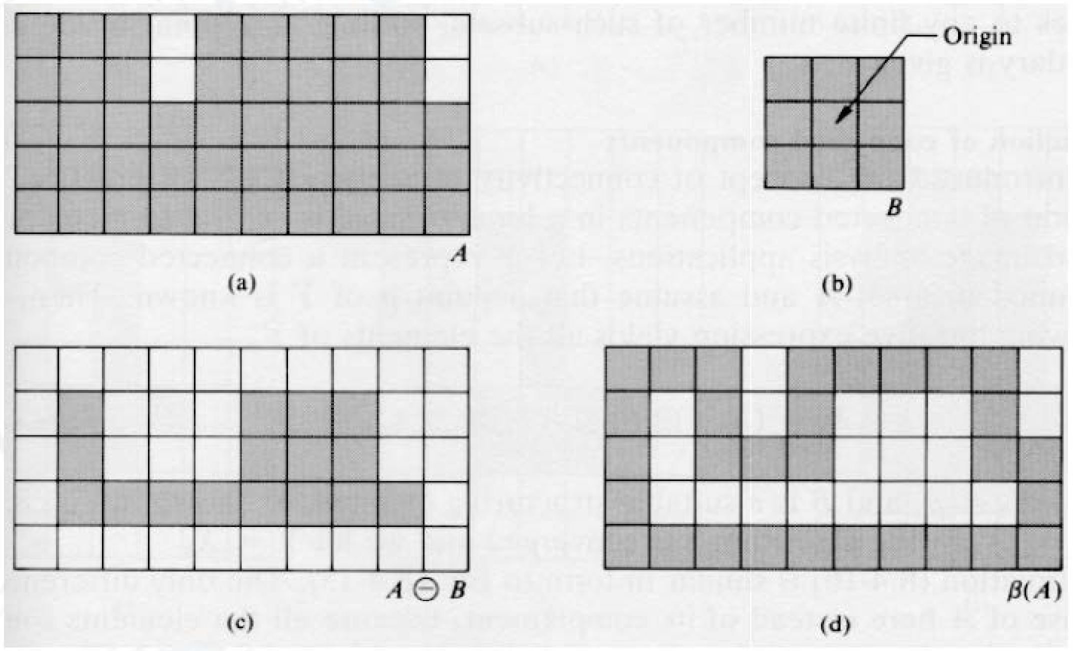


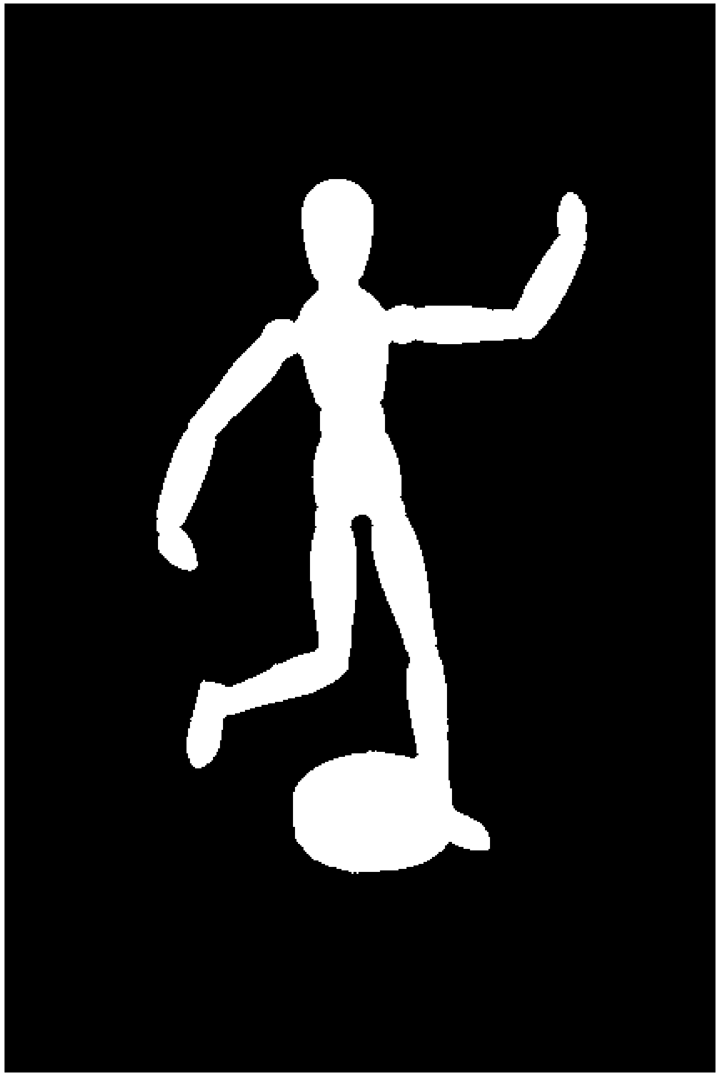
After closing

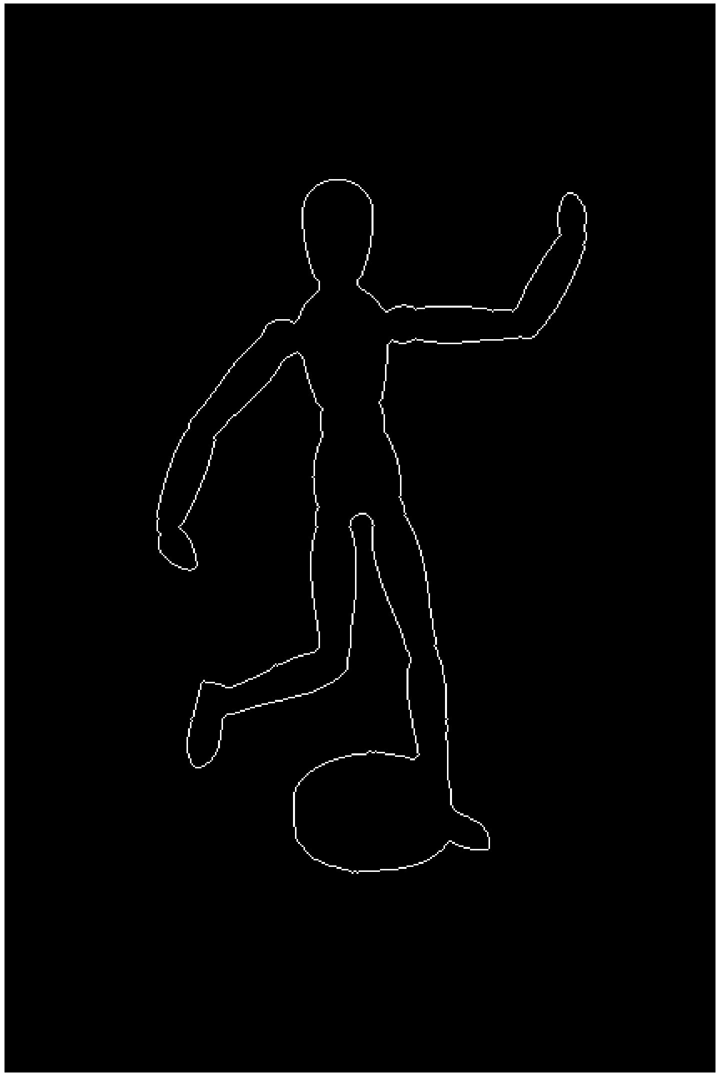
Applications

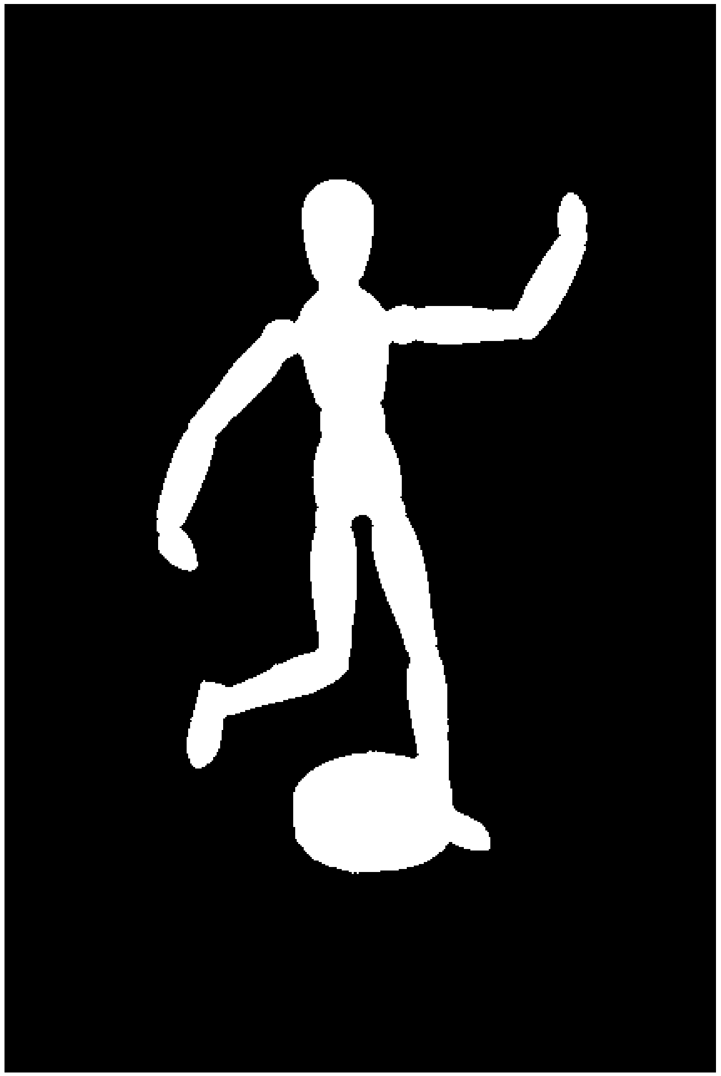
- **Boundary extraction (邊界的萃取)**

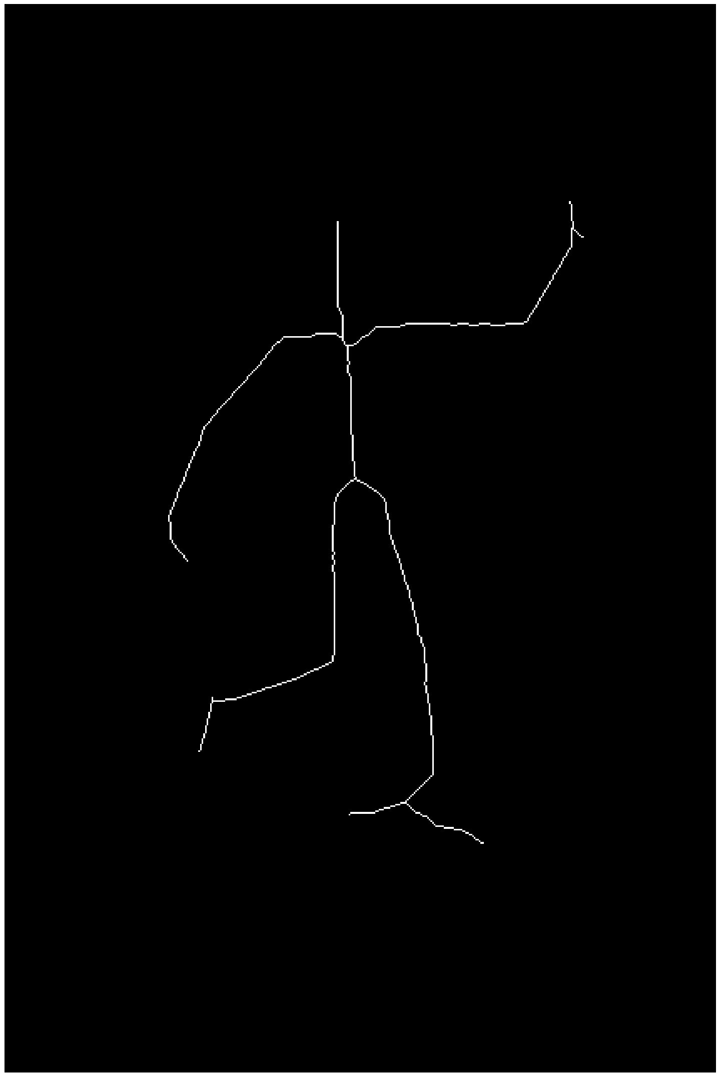
$$\beta(A) = A - (A \ominus B)$$




Binary segmentation


After boundary extraction

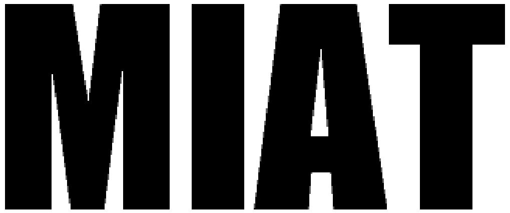


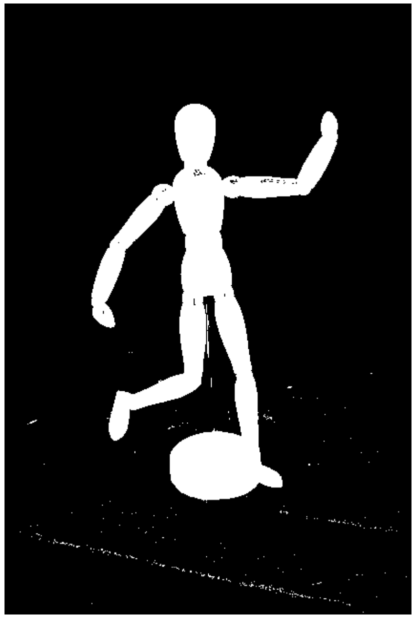


Repeatedly run erosion, stop when 1-pixel thick

Homework (plus)

- #6.1
  - 實做dilation
- #6.2
  - 實做erosion





```
#include "array.h"
/*-------------------------------------------------------------*/
//             Binary morphological image processing
//                        MIAT Lab
//               CHEN Ching-Han，2004.11.18
/*-------------------------------------------------------------*/
#define OBJ 0    // object
#define BG 255  // background
#define NumPtSet  5 // number of elements in structuring element
//    +
//  + + +  structuring element using N4
//    +
int seX[NumPtSet]={0,-1,0,1,0},seY[NumPtSet]={-1,0,0,0,1};

void clearImage(uc2D &ima)
{
  for(int i=0;i<nr;i++)for(int j=0;j<nc;j++)ima.m[i][j]=BG;
}
```

```
//-------------------------------------------------------------//
//              Binary Mathematical Morphology
//
//-----------------------------------------------------------------//
void binaryDilation(uc2D &im1, uc2D &im2)
{
  for(int i=0;i<nr;i++)for(int j=0;j<nc;j++)im2.m[i][j]=im1.m[i][j];
  for(int i=1;i<nr-1;i++)for(int j=1;j<nc-1;j++)
  {
    if(im1.m[i][j]==BG)
    for(int k=0;k<NumPtSet;k++)if(im1.m[i+seY[k]][j+seX[k]]==OBJ)
    {
      im2.m[i][j]=OBJ;
      break;
    }
  }
}
```

```
void binaryErosion(uc2D &im1, uc2D &im2)
{
   for(int i=0;i<nr;i++)for(int j=0;j<nc;j++)im2.m[i][j]=im1.m[i][j];
   for(int i=1;i<nr-1;i++)for(int j=1;j<nc-1;j++)
   {
     if(im1.m[i][j]==OBJ)
     for(int k=0;k<NumPtSet;k++)if(im1.m[i+seY[k]][j+seX[k]]==BG)
     {
       im2.m[i][j]=BG;
       break;
     }
   }
}
```