Binary image processing 2023/4/19

- **二進制圖像**

圖像在量化以獲得數字圖像之前包含連續的強度值，常用量化等級：256 (8-bits)、4096 (12-bits) （通常用於醫學圖像）
更多量化級別：更好的表示更大的存儲，而二值圖像是 2 級灰度量化（1bit）
內存和計算能力在早期是有限的，算法要很好理解，需要更少的內存和更快的執行時間和易於分析。
常應用於識別傳送帶上的組件、識別文本與符號、確定物體的方向。
缺點：需要適當照明提高對比度，某些應用無法用兩個灰度來恢復訊息。

- **二進制圖像處理**

大小為矩形(m*n)，通常表示方式為目標像素為 1（白色）、背景為 0（黑色）。
二值圖像處理重點：二值圖像的形成、幾何特性、拓撲性質、對象識別。
將灰階化的圖像用特定門檻值將像素二值化得到二進制圖像，甚至是非連續區間或特定的灰度。

$$F_T[i,j]=\begin{cases}1 & \text{if } F[i,j]\le T\\ 0 & \text{otherwise}\end{cases}$$

$$F_T[i,j]=\begin{cases}1 & \text{if } T_1\le F[i,j]\le T_2\\ 0 & \text{otherwise}\end{cases}$$

$$F_T[i,j]=\begin{cases}1 & \text{if } F[i,j]\in Z\\ 0 & \text{otherwise}\end{cases}$$

圖像二值化的第一步通常是透過調整門檻值來取得特定目的之二值化圖像

- **圖像分割處理**

將圖像依照指定規則分割，每個子區塊都是原圖像素的子集合。

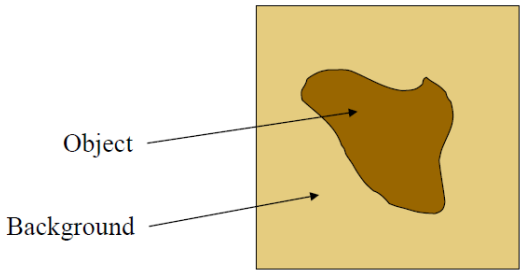
$$\cup_{i=1}^{k} P_i = \text{Entire image}$$
$$P_i \cap P_j = \varnothing,\ i \neq j$$

- 所有區塊總合為原圖
- 各區塊的範圍互相獨立
- 每個像素點都在分割出來的區塊內
- 在同一區塊的像素點都有共同的性質

- 幾何圖像

幾何特性通常為尺寸、位置、周長和方向，根據發出的信息做同物件判定而物件的面積由 0th moment 決定：

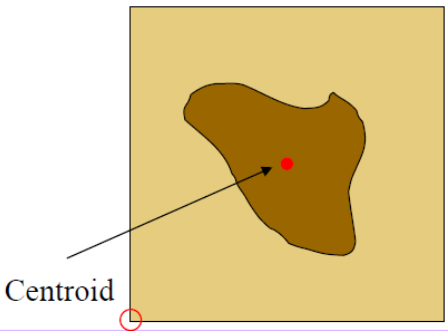
$$A = \sum_{i=1}^{n}\sum_{j=1}^{m} B(i,j)$$



包含整個物件的矩形稱為 bounding box，在物件內的點相對於雜訊更有辨識度，物件中心相當於物件的質心，中心由 1th moment 提供：

$$\bar{x} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} jB(i,j)}{\sum_{i=1}^{n}\sum_{j=1}^{m} B(i,j)}, \quad \bar{y} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} iB(i,j)}{\sum_{i=1}^{n}\sum_{j=1}^{m} B(i,j)}$$

$$\int xf(x)dx = \bar{x}\int f(x)dx$$



物件的方向依照需求所定義，但在極端情況（如圓形）的方向就難以判定，常見的判定方式為最小慣性軸的方向，並由 2nd moment 提供：

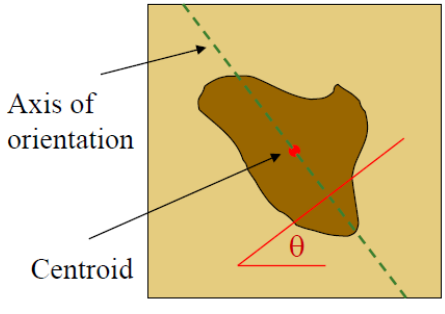
$$\theta = \frac{1}{2}\tan^{-1}\frac{M_{xy}}{M_{xx} - M_{yy}}$$

where the 2nd moments are

$$M_{xx} = \sum_{x}\sum_{y}(x-\bar{x})^2 B(x,y)$$

$$M_{xy} = \sum_{x}\sum_{y}2(x-\bar{x})(y-\bar{y})B(x,y)$$

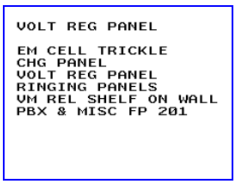
$$M_{yy} = \sum_{x}\sum_{y}(y-\bar{y})^2 B(x,y)$$

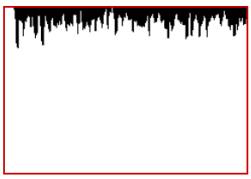


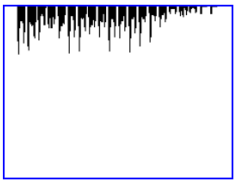
- 投影圖像

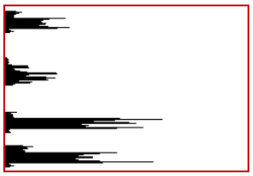
  投影是二進制圖像緊湊表示圖片，不同的圖像可能有相同的投影

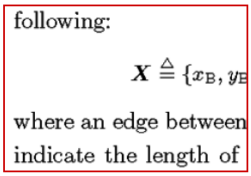
$$H[i]=\sum_{j=1}^{m} I(i,j), \qquad V[j]=\sum_{i=1}^{n} I(i,j)$$











$$A=\sum_{j=1}^{m} V[j]=\sum_{i=1}^{n} H[i], \quad \bar{x}=\frac{\sum_{j=1}^{m} jV[j]}{A}, \quad \bar{y}=\frac{\sum_{i=1}^{m} iH[j]}{A}$$

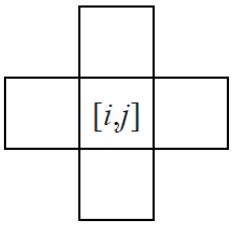
- Run-Length 編碼方式
  - Binary image:

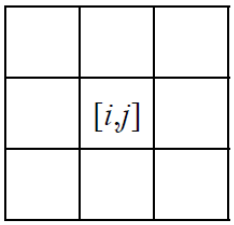
1	1	1	0	0	0	1	1	0	0	0	1	1	1	1	0	1	1	0	1	1	1
0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

  - Start and length of 1 runs:
    - (1,3) (7,2) (12,4) (17,2) (20,3)
    - (5,13) (19,4)
    - (1,3) (17,6)
  - Length of 1 and 0 runs:
    - 3,3,2,3,4,1,2,1,3
    - 0,4,13,1,4
    - 3,13,6

- 二進制圖像關聯方式

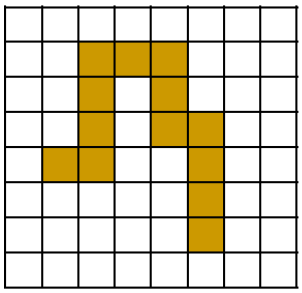
4 連通　　8 連通

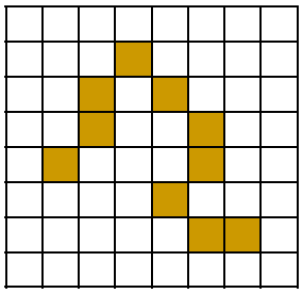




- 路徑

4 連通　　8 連通





- 周長

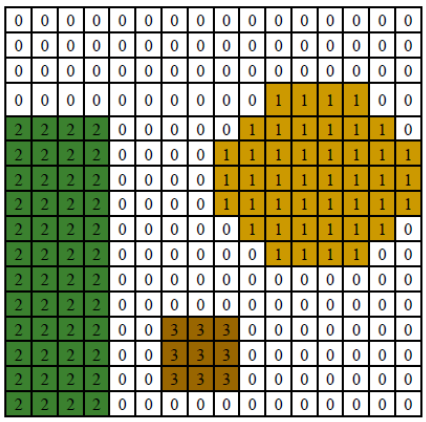
$$P_4 = \{(r,c) \in R \mid N_8(r,c) - R \neq \varnothing\}$$
$$P_8 = \{(r,c) \in R \mid N_4(r,c) - R \neq \varnothing\}$$
$$P = \langle (r_0,c_0), \ldots, (r_{k-1},c_{k-1}) \rangle$$
$$|P| = |\{k \mid (r_{k+1},c_{k+1}) \in N_4(r_k,c_k)\}| + 2^{1/2} |\{k \mid (r_{k+1},c_{k+1}) \in (N_8(r_k,c_k) - N_4(r_k,c_k))\}|$$

- 圓的判定

$C_1 = |P|^2 / A$　越小越接近真圓

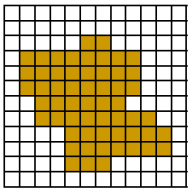
$C_2 = \mu_R / \sigma_R$　越大越接近真圓

$$\mu_R = \frac{1}{K}\sum_{k=0}^{K-1} \| (r_k,c_k) - (\bar{r},\bar{c}) \| \quad \sigma_R = \left(\frac{1}{K}\sum_{k=0}^{K-1} [\| (r_k,c_k) - (\bar{r},\bar{c}) \| - \mu_R]^2 \right)^{\frac{1}{2}}$$



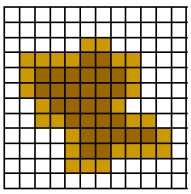
region number	region area	row of center	column of center	perimeter length	circularity 1	circularity 2	radius mean	radius variance
1	44	6	11.5	21.2	10.2	15.4	3.33	.05
2	48	9	1.5	28	16.3	2.5	3.80	2.28
3	9	13	7	8	7.1	5.8	1.2	0.04

- 邊界



S 的邊界是 S 中具有 4 個鄰域的像素集，邊界通常用 S' 表示

- 內飾



內部是 S 的不在其邊界內的像素集，S 的內部是 (S － S')

- 環繞

區域 T 包圍區域 S（或者 S 在 T 內部），如果從 S 的任意一點到圖片邊界的任何 4 條路徑都必須與 T 相交

- 物件標記

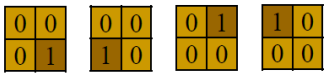
利用遞迴搜索：

掃描圖像找到未標記的 1 像素並為其分配新標籤 L
遞歸地為它的所有 1 個鄰居分配一個標籤 L
如果沒有更多未標記的 1 個像素，則停止
回到第一步，直到所有像素都被遍歷過

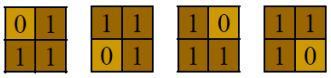
利用有序搜索：

掃描圖像，如果上與左都沒有 label：給予新的 label 值
其中之一有 label：給予有 label 的值
都有 label 並且相同：給予相同的 label 值
都有 label 但不相同：紀錄不同的 label 為相同物件，給予其一 label

- 計算物件數量

E 結構 

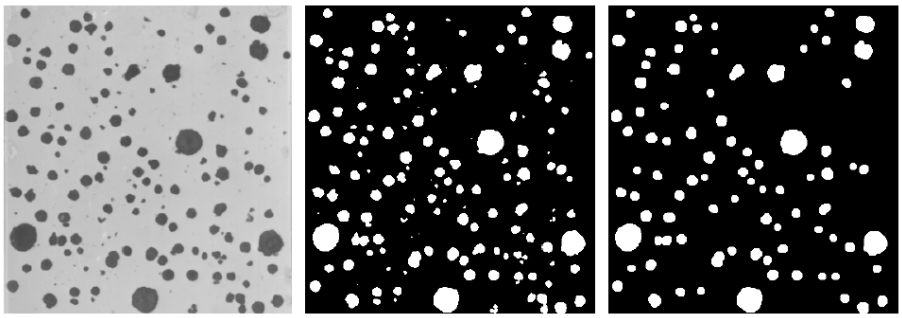
遍歷整張圖像，計算 E、I 結構數量

I 結構 

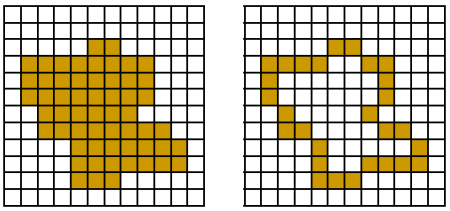
物件數量 = **(E – I) / 4**

- 過濾圖像方法

利用大小，大於指定面積大小的才算物件

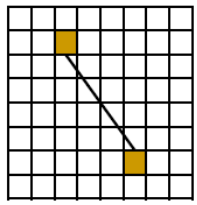


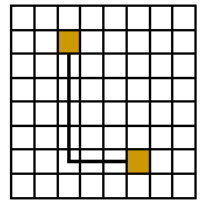
- Region Boundary
  - Boundary-Following Algorithm
    - Find a starting pixel $s \in S$ for the region via raster scan
    - Let the current pixel in boundary tracking be denoted by c, set $c = s$ and let the 4-neighbor to the west of s be $b \in \hat{S}$
    - Let the eight 8-neighbors of c starting with b in clockwise order be $n_1, n_2, \ldots, n_8$. Find n_i, for the first i that is in S
    - Set $c = n_i$ and $b = n_{i-1}$
    - Repeat above

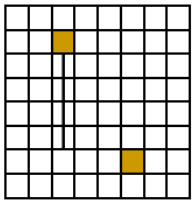


- 距離計算方式
  - Euclidean: $d_{\text{Euclidean}}([i_1, j_1],[i_2, j_2]) = \sqrt{(i_1 - i_2)^2 + (j_1 - j_2)^2}$
  - City-block: $d_{\text{city}}([i_1, j_1],[i_2, j_2]) = |i_1 - i_2| + |j_1 - j_2|$
  - Chessboard: $d_{\text{chess}}([i_1, j_1],[i_2, j_2]) = \max(|i_1 - i_2|, |j_1 - j_2|)$

Euclidean distance　　City-block distance　　Chessboard distance







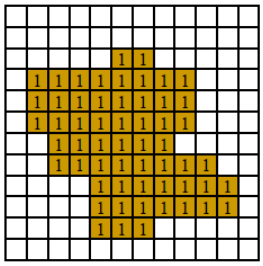
		3				
	$\sqrt{8}$	$\sqrt{5}$	2	$\sqrt{5}$	$\sqrt{8}$	
	$\sqrt{5}$	$\sqrt{2}$	1	$\sqrt{2}$	$\sqrt{5}$	
3	2	1	0	1	2	3
	$\sqrt{5}$	$\sqrt{2}$	1	$\sqrt{2}$	$\sqrt{5}$	
	$\sqrt{8}$	$\sqrt{5}$	2	$\sqrt{5}$	$\sqrt{8}$	
		3				

		3				
	3	2	3			
3	2	1	2	3		
3	2	1	0	1	2	3
3	2	1	2	3		
	3	2	3			
		3				

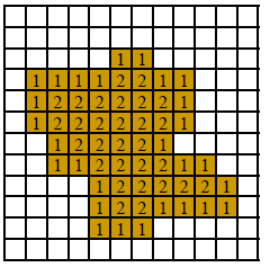
3	3	3	3	3	3	3
3	2	2	2	2	2	3
3	2	1	1	1	2	3
3	2	1	0	1	2	3
3	2	1	1	1	2	3
3	2	2	2	2	2	3
3	3	3	3	3	3	3

- 點與物件外的距離

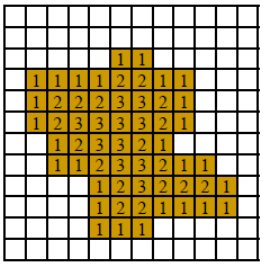
範例採用 City-block 距離方式，先將物件內的點標註為 1，依序往物件內以 4 連通依序判斷並賦予累加標註值。



0th pass

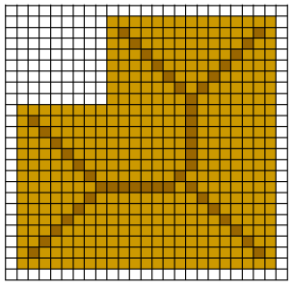


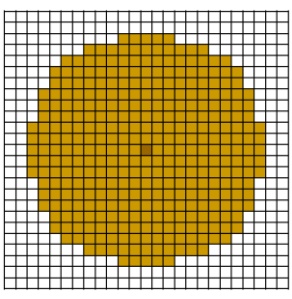
1st pass

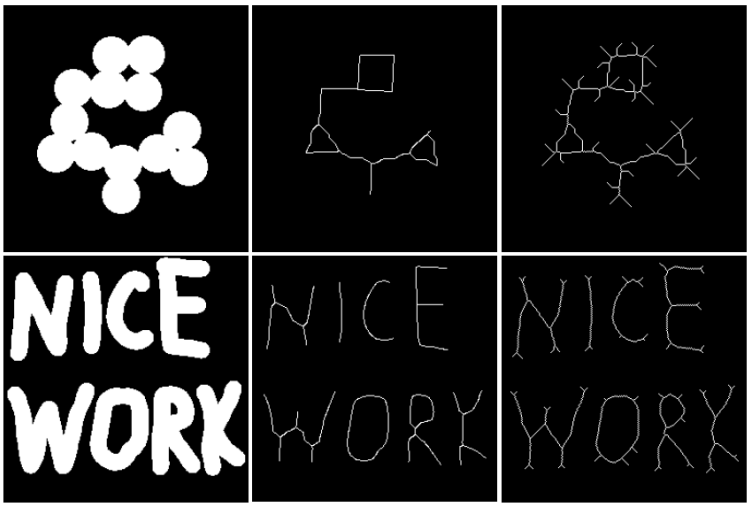


2nd pass

- 中軸判定







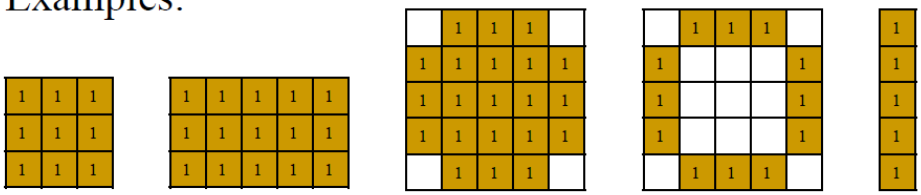
- 擴張和縮小

  擴張：將 1 附近為 0 的值變更為 1
  縮小：將 0 附近為 1 的值變更為 0
  “附近”的判定可以是 4 連通或 8 連通，依照需求決定
  擴張 n 次再縮小 n 次，並不一定會回到原始的形狀
  - $(S^m)^{-n} \neq (S^{-n})^m \neq S^{(m-n)}$
  - $S \subset (S^k)^{-k}$
  - $S \supset (S^{-k})^k$

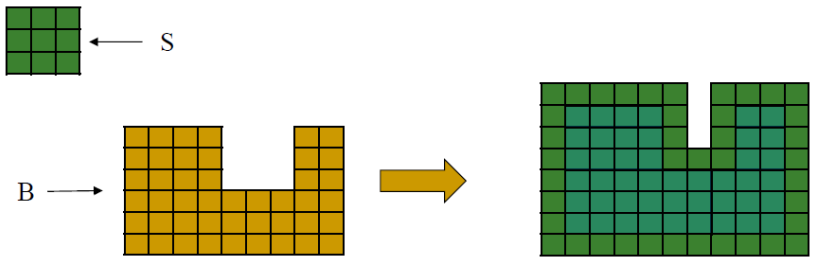
- 形態學

  依照指定的形狀對物件做範圍判定

  - Examples:



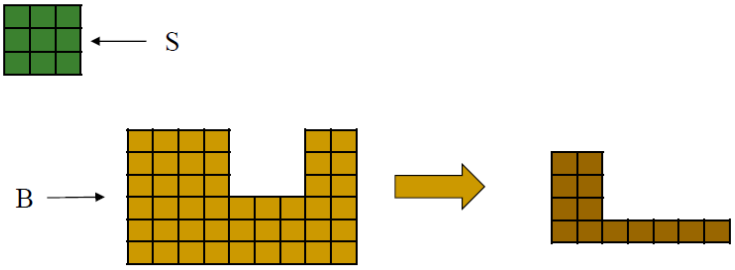
  ➢ 膨脹 dilation
  將物件擴大，可以看成從邊界以型態 S 走一遍，並將型態 S 碰過的地方都設為 1。



- 腐蝕 erosion

  將物件縮小，型態 S 在物件內走過所有的點（前提是他走的過去），最後求出型態 S 中心點走過的像素。

$$B \ominus S = \{x \mid S_x \subseteq B\} = \{b \mid b + s \in B, \forall s \in S\}$$

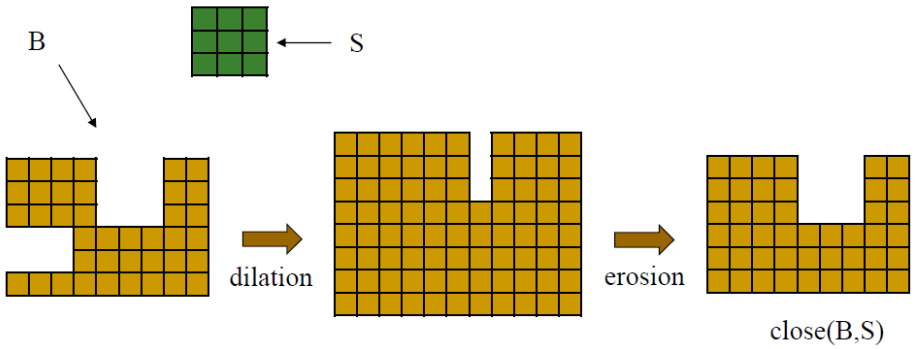


- 閉運算 closing

  (dilation 再 erosion)

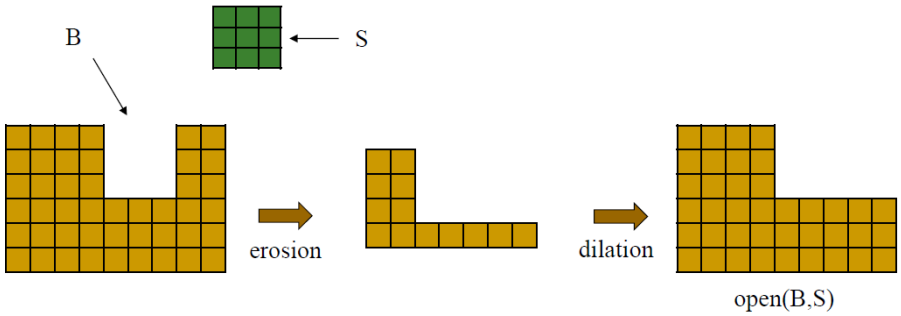
  填補狹窄的縫隙、孔洞和小裂縫

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

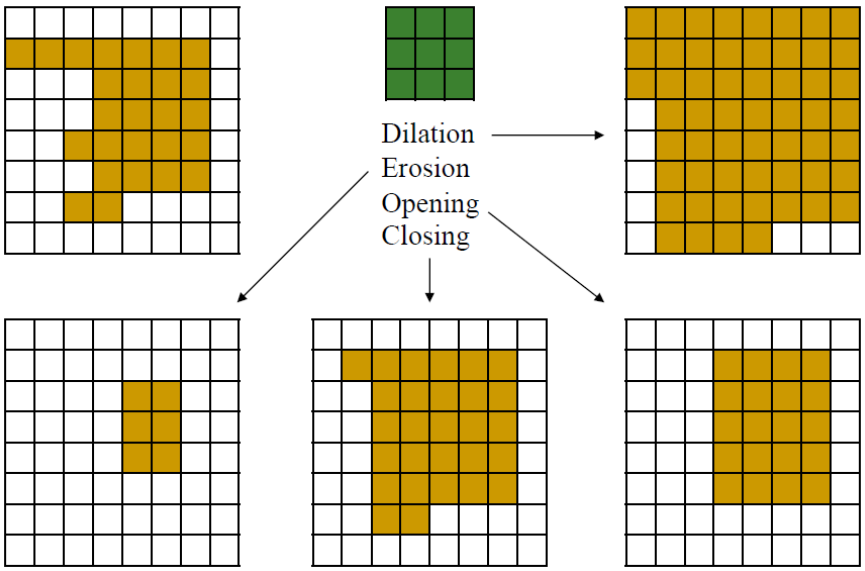


- 開運算 opening
  (erosion 再 dilation)
  平滑輪廓、擴大狹窄的縫隙、消除微小的突起

$$B \circ S = (B \ominus S) \oplus S = \cup \{S_x \mid S_x \subseteq B\}$$

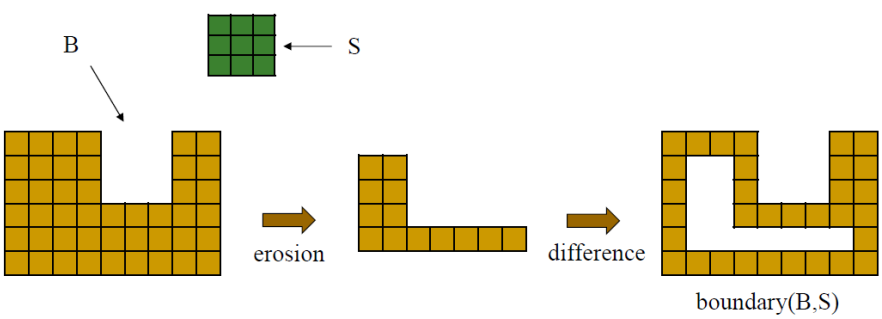


對同一物件做不同處理：



- 尋找邊界

$$boundary(B,S) = B - B \ominus S$$



- 填滿區塊
  - Let $C_0 = p$
  - Calculate

$$C_k = (C_{k-1} \oplus S) \cap B^c, \text{ for } k = 1,2, \ldots$$

  - Stop when $C_k = C_{k-1}$
  - C_k is the interior of B