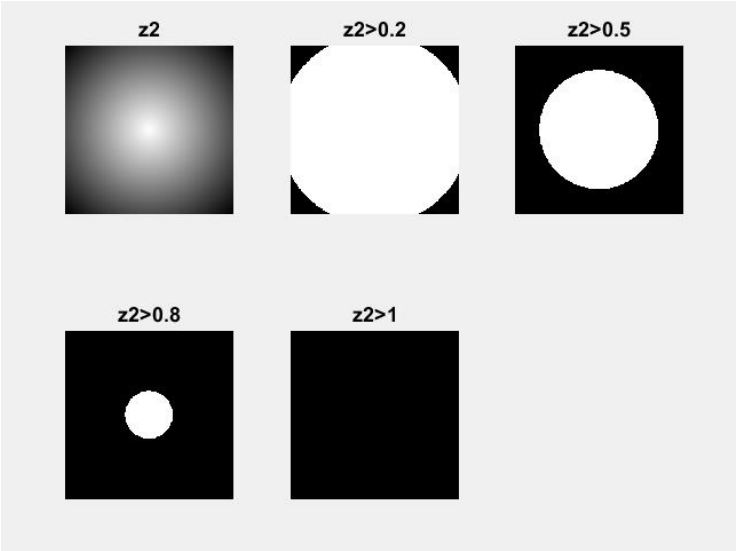
電機三 410587040 詹閎棋

HW4

CH9

2. (實驗)

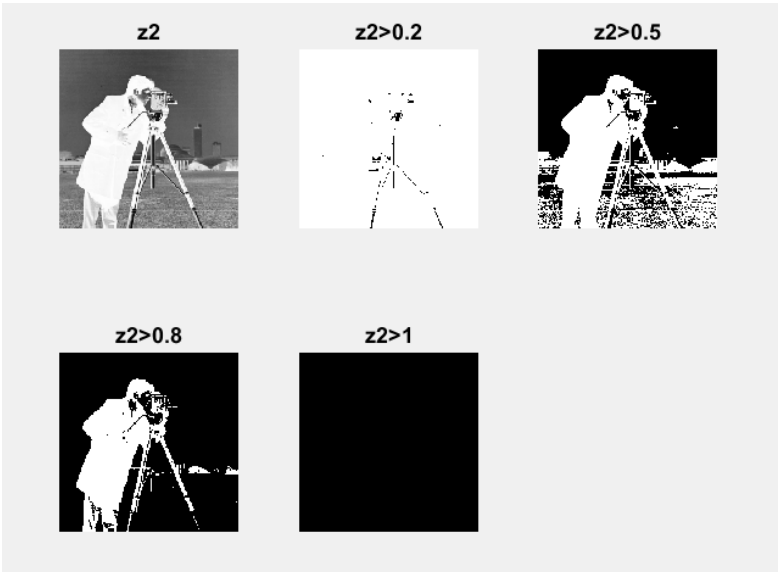
```
[x,y]=meshgrid(1:256,1:256);
z=sqrt((x-128).^2+(y-128).^2);
z2=1-mat2gray(z);
subplot(2,3,1),imshow(z2);
title('z2');
subplot(2,3,2),imshow(z2>0.2);
title('z2>0.2');
subplot(2,3,3),imshow(z2>0.5);
title('z2>0.5');
subplot(2,3,4),imshow(z2>0.8);
title('z2>0.8');
subplot(2,3,5),imshow(z2>1);
title('z2>1');
```



當閥值增加時，白色部分逐漸縮小，閥值增加到 1 時，白色部分消失。

3. (實驗)

```
c=imread('cameraman.png');
z2=1-mat2gray(c);
subplot(2,3,1),imshow(z2);
title('z2');
subplot(2,3,2),imshow(z2>0.2);
title('z2>0.2');
subplot(2,3,3),imshow(z2>0.5);
title('z2>0.5');
subplot(2,3,4),imshow(z2>0.8);
title('z2>0.8');
subplot(2,3,5),imshow(z2>1);
title('z2>1');
```



使用影像 cameraman.png。

5. (實驗)

```
n=im2uint8(imread('nicework.png'));
c=imread('cameraman.png');
m=imlincomb(0.5,c,1,n);
imshow(m);
```

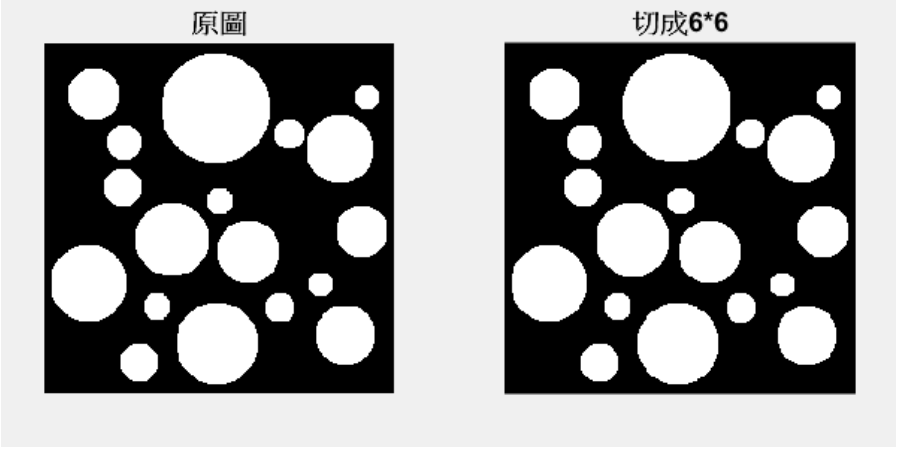
將 nicework.png 的閥值調高，即可找出文字。

6. (實驗)

```
1 —    n=im2uint8(imread('nicework.png'));
2 —    c=imread('cameraman.png');
3 —    n=im2double(n);
4 —    c=im2double(c);
5 —    m=c.*(n==0);
6 —    imshow(m);
```

7. (討論)

```
t=imread('circles.png');
figure(1),imshow(t);
[r,c]=size(t);
[x,y]=meshgrid(1:256,1:256);
t2=double(t).*(x+y)/512;
t3=im2uint8(t2);


thresh=@(x)imbinarize(x.data);
figure(2),imshow(blockproc(t3,[r/6,c/6],thresh));
% r、c決定分幾塊
```



將長、寬各切成 6 等分，共 36 區塊，可完整擷取出來。

8. (討論)

```
im=200*ones(10,10);
im(3:5,3:8)=50;
im(6:8,4:7)=50;
s=im+round(9*randn(10,10))

rx=[1 0 0;0 -1 0;0 0 0];
ry=[0 1 0;-1 0 0;0 0 0];
srx=imfilter(s,rx);
sry=imfilter(s,ry);
edge_r=uint8(sqrt(double(srx).^2+double(sry).^2));
edge_r

px=[-1 0 1;-1 0 1;-1 0 1];
py=px';
spx=imfilter(s,px);
spy=imfilter(s,py);
edge_p=uint8(sqrt(double(spx).^2+double(spy).^2));
edge_p

sx=[-1 0 1;-2 0 2;-1 0 1];
sy=[-1 -2 1;0 0 0;1 2 1];
ssx=imfilter(s,sx);
ssy=imfilter(s,sy);
edge_s=uint8(sqrt(double(ssx).^2+double(ssy).^2));
edge_s
```

```
26
27 —     lap=fspecial('laplacian',0);
28 —     s_lap=imfilter(s,lap);
29 —     s_lap
30
31 —     lap=fspecial('laplacian',0);
32 —     sz=edge(s,'zerocross',[],lap);
33 —     sz
34
```

```
s =

   208   194   194   183   204   205   200   197   192   193
   200   204   190   187   195   207   188   199   196   196
   201   206    41    48    54    55    66    45   193   218
   210   197    51    37    57    33    55    43   199   204
   190   180    49    47    60    54    61    74   193   209
   202   199   191    45    45    60    45   202   201   203
   200   200   205    50    50    58    47   218   184   223
   215   203   190    48    44    50    43   191   218   207
   198   218   206   207   192   204   197   190   189   188
   193   200   192   203   196   209   202   201   194   191
```

原矩陣

```
edge_r =

  10×10 uint8 matrix

   208   255   255   255   255   255   255   255   255   255
   255     7    11    10    21    10    18     9     7     5
   255     7   164   204   198   207   194   195   151    22
   255     6   220     5    19    21    33    25   215    22
   255    31   196    13    25    27    28    26   195    15
   255    24   150   144    15     9     9   144   127    13
   255     2    11   213     7    16    18   232    25    29
   255    15    10   210     6    14    15   227     7    24
   255     6    28   159   217   214   218   147    28    35
   255    25    27    15    16    19    12    13    13     6
```

Roberts 方法

```
edge_p =

  10×10 uint8 matrix

   255   255   255   255   255   255   255   255   255   255
   255   236   255   255   255   255   255   255   208   255
   255   255   255   255   255   255   255   255   255   255
   255   255   255    33    11    11    30   255   255   255
   255   255   255   187    29     5   246   255   255   255
   255   237   255   255    30    20   255   255   205   255
   255    35   255   255    26    14   255   255    24   255
   255    21   255   255   255   255   255   255    61   255
   255    29   224   255   255   255   255   215    30   255
   255   255   255   255   255   255   255   255   255   255
```

Prewitt 方法

```
edge_s =

  10×10 uint8 matrix

   255   255   255   255   255   255   255   255   255   255
   255   255   213   168   188   184   179   145   255   255
   255   255   255   191   188   243   199   255   255   255
   255   255   255   125   120   131   112   255   255   255
   255   255   255   255    81   142   255   255   255   255
   255   255   255   255   105   106   255   255   255   255
   255   255   255   255   116    67   255   255   255   255
   255   255   255   255   255   255   255   255   255   255
   255   255   255   255   255   255   255   255   255   255
   255   255   255   255   255   255   255   255   255   255
```

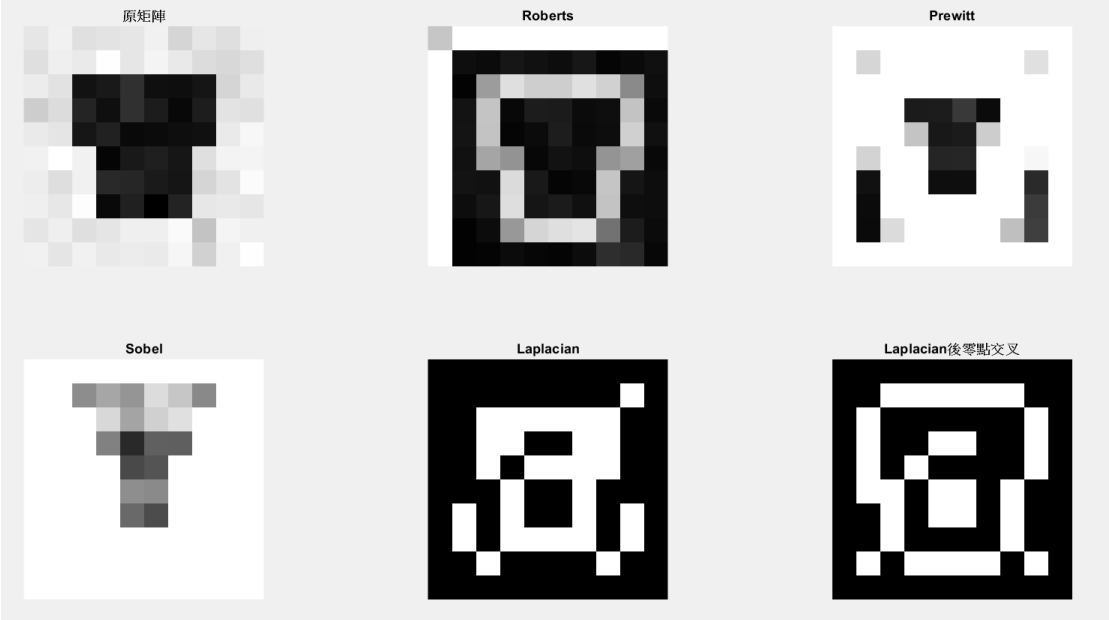
Sobel 方法

```
s_lap =

  -438  -170  -209  -147  -233  -209  -210  -197  -182  -384
  -187   -26  -134  -132  -128  -185   -80  -170    -4  -177
  -188  -181   331   127   139   140    79   321  -114  -279
  -252  -141   120    55   -44    89   -17   201  -163  -190
  -168   -85   273     3   -37    -2   -16   203   -89  -236
  -219   -23  -266   153    35   -38   190  -270   -22  -179
  -183     7  -189   148    -3   -25   176  -248   124  -298
  -259    11   -98   299   164   149   313   -95  -101  -199
  -166   -65   -17  -179  -117  -168  -149    18    34  -165
  -374  -197  -159  -217  -180  -234  -201  -218  -195  -382
```

Laplacian 方法

```
sz =

  10×10 logical array

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

零點交叉方法



由實驗看出，Roberts 方法產生最好的結果。

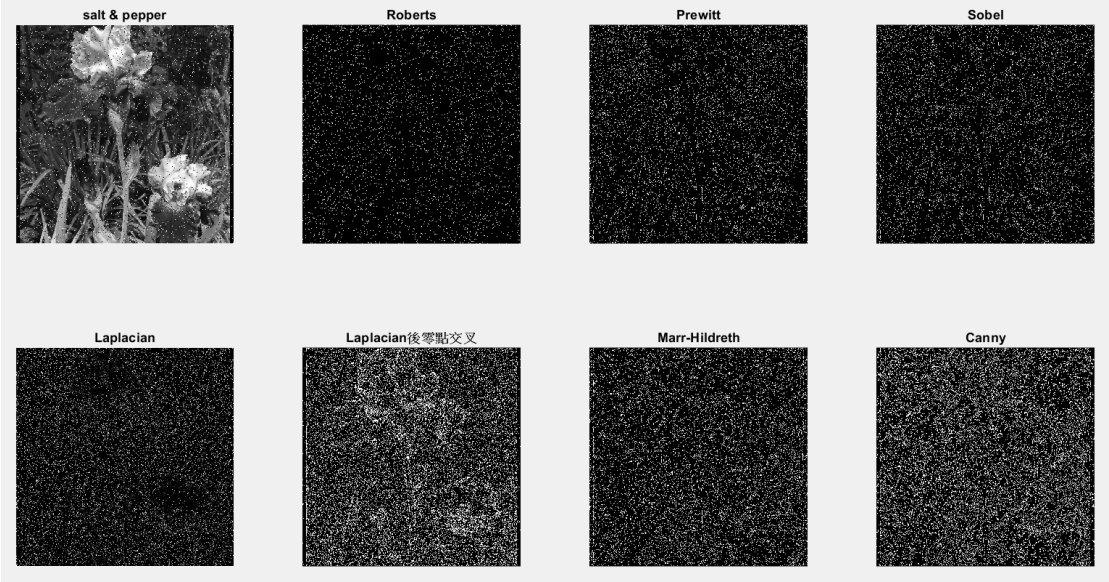
10. (實驗)

```
c=imread('cameraman.png');
subplot(2,4,1),imshow(c);
title('原圖');
edge_r=edge(c,'roberts');
subplot(2,4,2),imshow(edge_r);
title('Roberts');
edge_p=edge(c,'prewitt');
subplot(2,4,3),imshow(edge_p);
title('Prewitt');
edge_s=edge(c,'sobel');
subplot(2,4,4),imshow(edge_s);
title('Sobel');
lap=fspecial('laplacian',0);
c_lap=imfilter(c,lap);
subplot(2,4,5),imshow(c_lap);
title('Laplacian');
lap=fspecial('laplacian',0);
cz=edge(c,'zerocross',[],lap);
subplot(2,4,6),imshow(cz);
title('Laplacian後零點交叉');
log=fspecial('log',13,2);
cmh=edge(c,'zerocross',[],log);
subplot(2,4,7),imshow(cmh);
title('Marr-Hildreth');
cc=edge(c,'canny');
subplot(2,4,8),imshow(cc);
```

根據實驗結果，使用 Canny 這個邊緣偵測方法所得到的結果最好。

13. (實驗)

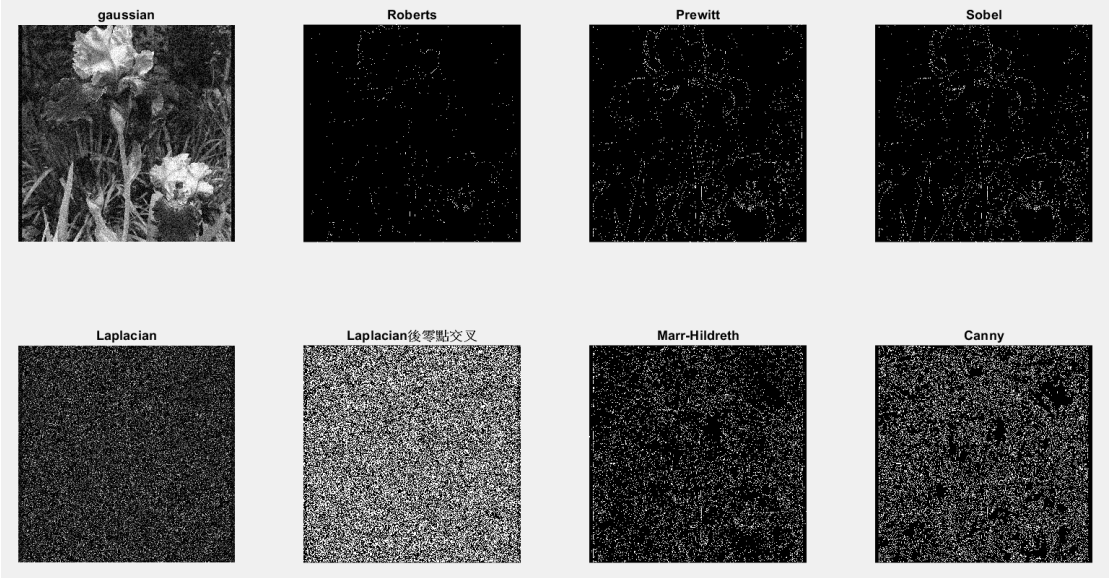
C1: salt & pepper



(a) 在有 salt & pepper 雜訊的情況下，Laplacian 後零點交叉得到最好的結果。

(b) 在有 salt & pepper 雜訊的情況下，Marr-Hildreth 得到最差的結果。

C2: gaussian



(a) 在有 gaussian 雜訊的情況下，Sobel 得到最好的結果。

(b) 在有 gaussian 雜訊的情況下，Laplacian 後零點交叉得到最差的結果。

CH10

1. (實驗)

```
35 —    td1=imdilate(A1,B1);
36 —    td2=imdilate(A2,B2);
37 —    td3=imdilate(A3,B3);
38      % 膨脹
39
40 —    re1=imerode(A1,B1);
41 —    re2=imerode(A2,B2);
42 —    re3=imerode(A3,B3);
43      % 侵蝕
44
45 —    open1=imopen(A1,B1);
46 —    open2=imopen(A2,B2);
47 —    open3=imopen(A3,B3);
48      % 開啟
49
50 —    close1=imclose(A1,B1);
51 —    close2=imclose(A2,B2);
52 —    close3=imclose(A3,B3);
53      % 關閉
```

td1 | td2 | td3 | re1 | re2 | re3 | open1 | open2 | open3

8x8 double

	1	2	3	4	5	6	7	8
1	0	0	0	1	1	1	1	0
2	0	0	1	1	1	1	1	1
3	0	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	0
7	1	1	1	1	1	1	0	0
8	0	1	1	1	1	0	0	0

A1⊕B1

td1 | td2 | td3 | re1 | re2 | re3 | open1 | open2 | open3

8x8 double

	1	2	3	4	5	6	7	8
1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1	1

A2⊕B2

td1 td2 td3 re1 re2 re3 open1 open2 open3

8x8 double

	1	2	3	4	5	6	7	8
1	0	0	0	0	1	1	0	-Inf
2	1	1	1	0	1	1	0	0
3	1	1	1	0	1	1	1	1
4	1	1	1	1	1	1	1	1
5	1	1	1	1	1	0	1	1
6	1	1	1	1	1	0	1	1
7	0	0	1	1	1	0	0	0
8	-Inf	0	1	1	1	0	0	0

A3 $\oplus$ B3

td1 td2 td3 re1 re2 re3 open1 open2 open3

8x8 double

	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	1	1	0	0
4	0	0	0	1	1	1	0	0
5	0	0	1	1	1	0	0	0
6	0	0	1	1	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0

A1 $\ominus$ B1

td1 td2 td3 re1 re2 re3 open1 open2 open3

8x8 double

	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0

A2 $\ominus$ B2

td1 td2 td3 re1 re2 re3 open1 open2 open3

8x8 double

	1	2	3	4	5	6	7	8
1	0	0	0	0	1	1	0	Inf
2	1	0	0	0	0	0	0	0
3	1	0	0	0	0	0	0	1
4	1	0	1	0	1	0	0	1
5	1	0	1	0	0	0	0	1
6	1	0	1	0	0	0	0	1
7	0	0	0	0	0	0	0	0
8	Inf	0	1	1	1	0	0	0

A3 $\ominus$ B3

td1 td2 td3 re1 re2 re3 open1 open2 open3

8x8 double

	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	0	0	0	0	1	1	0	0
3	0	0	0	1	1	1	1	0
4	0	0	1	1	1	1	1	0
5	0	1	1	1	1	1	0	0
6	0	1	1	1	1	0	0	0
7	0	0	1	1	0	0	0	0
8	0	0	0	0	0	0	0	0

A1 ◦ B1

td1 td2 td3 re1 re2 re3 open1 open2 open3

8x8 double

	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0

A2 ◦ B2

td1 td2 td3 re1 re2 re3 open1 open2 open3

8x8 double

	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	-Inf
2	0	0	0	0	0	1	1	0
3	0	1	0	1	0	0	1	0
4	0	1	0	0	0	0	1	0
5	0	1	0	1	0	1	1	0
6	0	1	0	1	0	0	0	0
7	0	1	1	1	0	0	0	0
8	-Inf	0	0	0	0	0	0	0

A3 ◦ B3

td1 td2 td3 re1 re2 re3 open1 open2 open3

8x8 double

	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	0	0	0	1	1	1	1	0
3	0	0	1	1	1	1	1	0
4	0	1	1	1	1	1	1	0
5	0	1	1	1	1	1	1	0
6	0	1	1	1	1	1	0	0
7	0	1	1	1	1	0	0	0
8	0	0	0	0	0	0	0	0

A1 · B1

td1 | td2 | td3 | re1 | re2 | re3 | open1 | open2 | open3 | close1 | clos

8x8 double

	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	0	0	0		
2	0	1	1	1	1	1	1	0		
3	0	1	1	1	1	1	1	0		
4	0	1	1	1	1	1	1	0		
5	0	1	1	1	1	1	1	0		
6	0	1	1	1	1	1	1	0		
7	0	1	1	1	1	1	1	0		
8	0	0	0	0	0	0	0	0		
9										

A2 · B2

td1 | td2 | td3 | re1 | re2 | re3 | open1 | open2 | open3

8x8 double

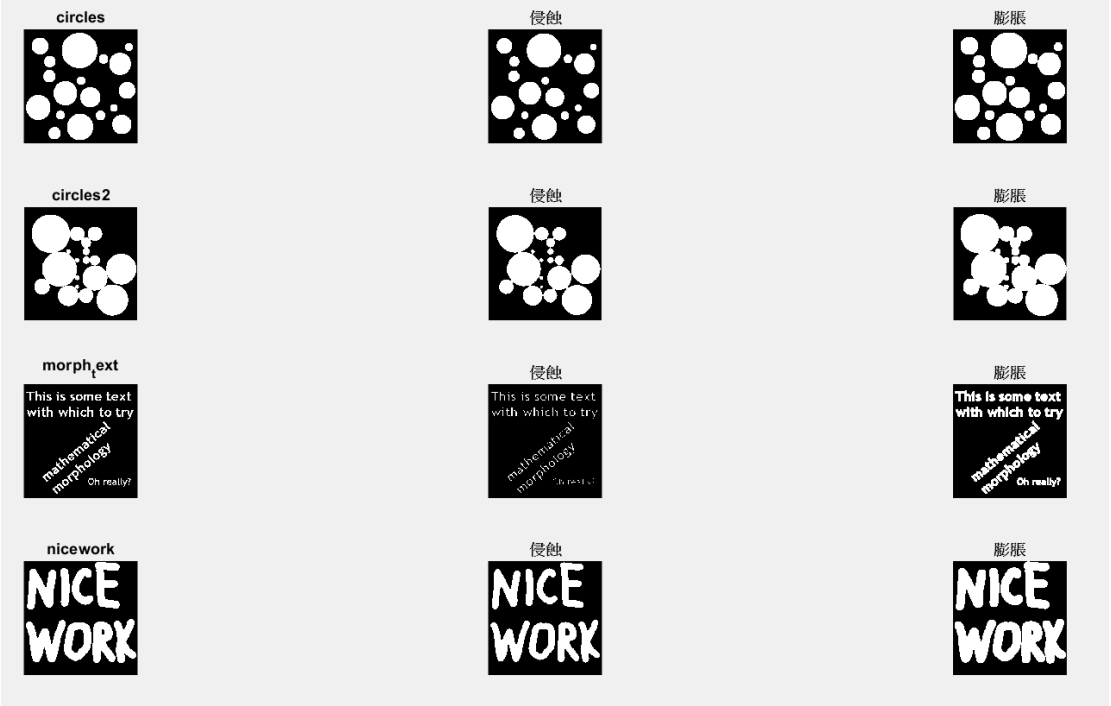
	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	1	1	0
3	0	1	1	1	0	1	1	0
4	0	1	1	1	0	1	1	0
5	0	1	1	1	0	1	1	0
6	0	1	1	1	0	0	0	0
7	0	1	1	1	0	0	0	0
8	0	0	0	0	0	0	0	0

A3 · B3

4. (實驗)

```
c1=imread('circles.png');
c2=imread('circles2.png');
m=imread('morph_text.png');
n=imread('nicework.png');

sq=[1 1 1;1 1 1;1 1 1]; %方型
cr=[0 1 0;1 1 1;0 1 0]; %十字

sqrec1=imerode(c1,sq);
sqrec2=imerode(c2,sq);
sqrem=imerode(m,sq);
sqren=imerode(n,sq);
crrec1=imerode(c1,cr);
crrec2=imerode(c2,cr);
crrem=imerode(m,cr);
crren=imerode(n,cr);
%侵蝕

sqtdc1=imdilate(c1,sq);
sqtdc2=imdilate(c2,sq);
sqtdm=imdilate(m,sq);
sqtdn=imdilate(n,sq);
crtdc1=imdilate(c1,cr);
crtdc2=imdilate(c2,cr);
```

```
25 -    crtdm=imdilate(m,cr);
26 -    crtdn=imdilate(n,cr);
27      %膨脹
28
29 -    subplot(4,3,1),imshow(c1);
30 -    title('circles');
31 -    subplot(4,3,2),imshow(sqrec1);
32 -    title('侵蝕');
33 -    subplot(4,3,3),imshow(sqtdc1);
34 -    title('膨脹');
35
36 -    subplot(4,3,4),imshow(c2);
37 -    title('circles2');
38 -    subplot(4,3,5),imshow(sqrec2);
39 -    title('侵蝕');
40 -    subplot(4,3,6),imshow(sqtdc2);
41 -    title('膨脹');
42
43 -    subplot(4,3,7),imshow(m);
44 -    title('morph_text');
45 -    subplot(4,3,8),imshow(sqrem);
46 -    title('侵蝕');
47 -    subplot(4,3,9),imshow(sqtdm);
48 -    title('膨脹');
49
50 -    subplot(4,3,10),imshow(n);
51 -    title('nicework');
52 -    subplot(4,3,11),imshow(sqren);
53 -    title('侵蝕');
54 -    subplot(4,3,12),imshow(sqtdn);
55 -    title('膨脹');
```

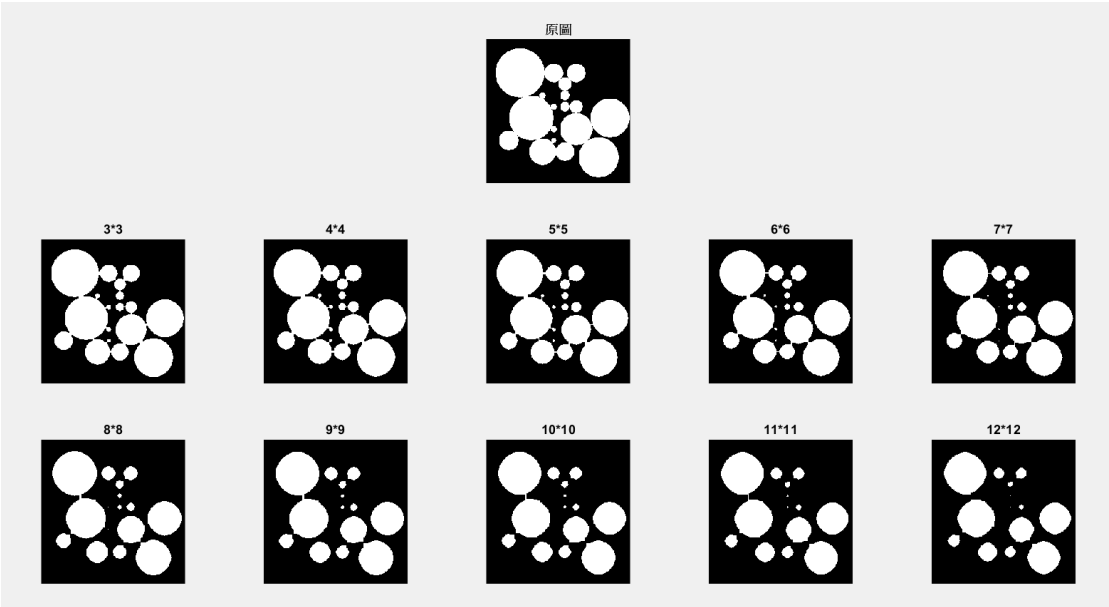


可以看出差異，且 morph_text 最明顯。

5. (實驗)

(a)

```
c=imread('circles2.png');
subplot(3,5,3),imshow(c);
title('原圖');

a3=ones(3);
ca3=imerode(c,a3);
subplot(3,5,6),imshow(ca3);
title('3*3');

a4=ones(4);
ca4=imerode(c,a4);
subplot(3,5,7),imshow(ca4);
title('4*4');

a5=ones(5);
ca5=imerode(c,a5);
subplot(3,5,8),imshow(ca5);
title('5*5');

a6=ones(6);
ca6=imerode(c,a6);
subplot(3,5,9),imshow(ca6);
title('6*6');

a7=ones(7);
ca7=imerode(c,a7);
subplot(3,5,10),imshow(ca7);
title('7*7');

a8=ones(8);
ca8=imerode(c,a8);
subplot(3,5,11),imshow(ca8);
title('8*8');

a9=ones(9);
ca9=imerode(c,a9);
subplot(3,5,12),imshow(ca9);
title('9*9');

a10=ones(10);
ca10=imerode(c,a10);
subplot(3,5,13),imshow(ca10);
title('10*10');

a11=ones(11);
ca11=imerode(c,a11);
subplot(3,5,14),imshow(ca11);
title('11*11');

a12=ones(12);
ca12=imerode(c,a12);
subplot(3,5,15),imshow(ca12);
title('12*12');
```

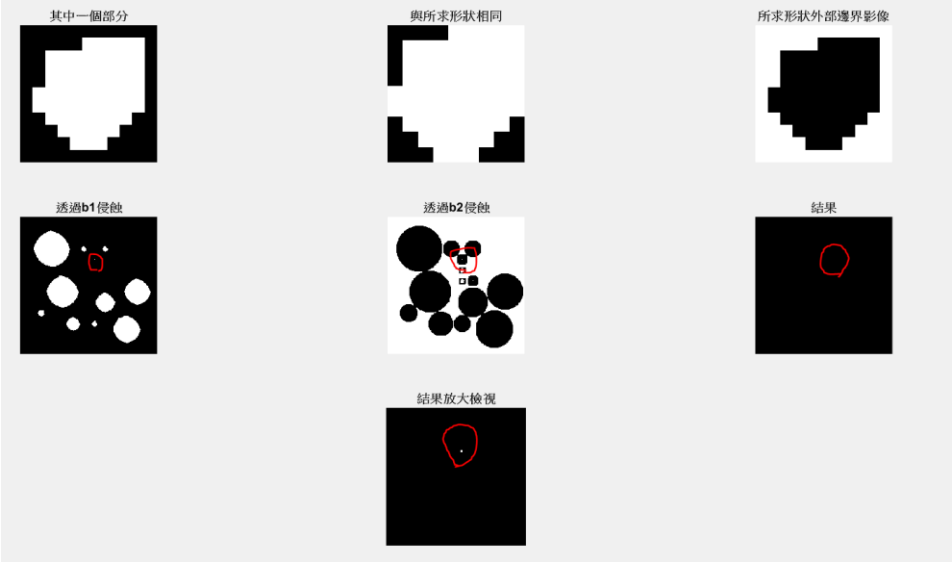


當方形結構元素加大到 12*12 時，影像全部分裂為無法連接的多個部分。

(b)

```
c=imread('circles2.png');

al2=ones(12);
cal2=imerode(c,al2);
cal2=mat2gray(cal2);
cc=cal2(75:85,135:145);

bl=[0 0 0 0 1 1 1 1 1 1
    0 1 1 1 1 1 1 1 1 1
    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
    1 1 1 1 1 1 1 1 1 1
    0 1 1 1 1 1 1 1 1 0
    0 0 1 1 1 1 1 0 0
    0 0 0 1 1 1 0 0 0];
b2=ones(11);
b2(2:10,2:10)=~bl;

cbl=imerode(cal2,bl);
cb2=imerode(~cal2,b2);
hit_or_miss=cbl&cb2;
[x,y]=find(hit_or_miss==1)
% 得出座標
cxy=hit_or_miss(60:124,105:170);
```

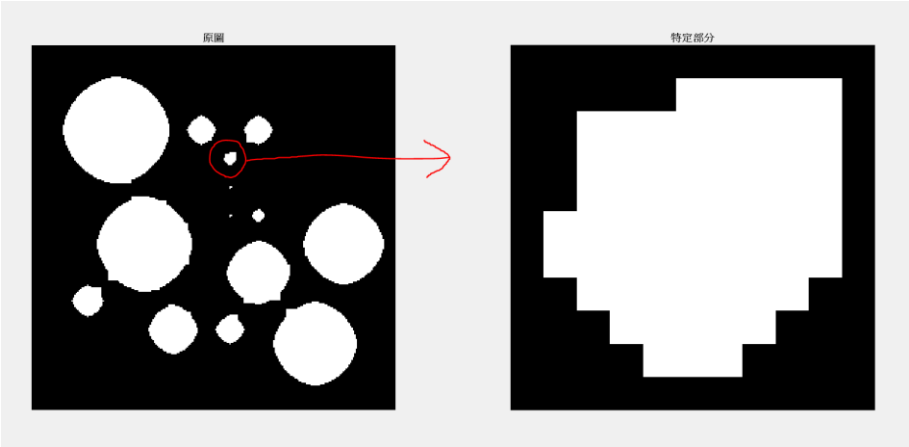
```
24      % 得出座標
25 -    cxy=hit_or_miss(60:124,105:170);
26
27
28 -    subplot(3,3,1),imshow(cc);
29 -    title('其中一個部分');
30 -    subplot(3,3,2),imshow(b1);
31 -    title('與所求形狀相同');
32 -    subplot(3,3,3),imshow(b2);
33 -    title('所求形狀外部邊界影像');
34 -    subplot(3,3,4),imshow(cb1);
35 -    title('透過b1侵蝕');
36 -    subplot(3,3,5),imshow(cb2);
37 -    title('透過b2侵蝕');
38 -    subplot(3,3,6),imshow(hit_or_miss);
39 -    title('結果');
40 -    subplot(3,3,8),imshow(cxy);
41 -    title('結果放大檢視');
```




```
x =

    80


y =

   140
```

(x , y)座標為(80 , 140)。

(c)



獨立出該特定部分。

6. (實驗)

(a)

```
c=imread('circles2.png');

al2=ones(12);
cal2=imerode(c,al2);
cal2=mat2gray(cal2);
cc=cal2(75:85,135:145);
cc;
b=[1 0 1
    0 1 0
    1 0 1];
cd=imdilate(cc,b);
cd_ext=cd&~cc;
cd_ext
imshow(cd_ext);
% 求出外部邊界
```

計算出其外部邊界。

(b)

```
bl=[0 0 0 0 1 1 1 1 1
    0 1 1 1 1 1 1 1 1
    0 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 1 1 1 1 1 1 1
    0 1 1 1 1 1 1 1 0
    0 0 1 1 1 1 1 0 0
    0 0 0 1 1 1 0 0 0];
b2=cd_ext;
ccbl=imerode(cc,bl);
ccb2=imerode(~cc,b2);
hit_or_miss=ccbl&ccb2;
[x,y]=find(hit_or_miss==1)
```

```
x =

     6


y =

     6
```

邊界內的一個像素(x , y)=(6 , 6)。

(c)

```
bl=[0 0 0 0 1 1 1 1 1
    0 1 1 1 1 1 1 1 1
    0 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 1 1 1 1 1 1 1
    0 1 1 1 1 1 1 1 0
    0 0 1 1 1 1 1 0 0
    0 0 0 1 1 1 0 0 0];
b2=cd_ext;
ccbl=imerode(cc,bl);
ccb2=imerode(~cc,b2);
hit_or_miss=ccbl&ccb2;
[x,y]=find(hit_or_miss==1)

sq=ones(3);
ccb=cc&~imerode(cc,sq);
figure(3),imshow(ccb);
ccf=imfill(ccb,[6,6],sq);
figure(4),imshow(ccf);
```

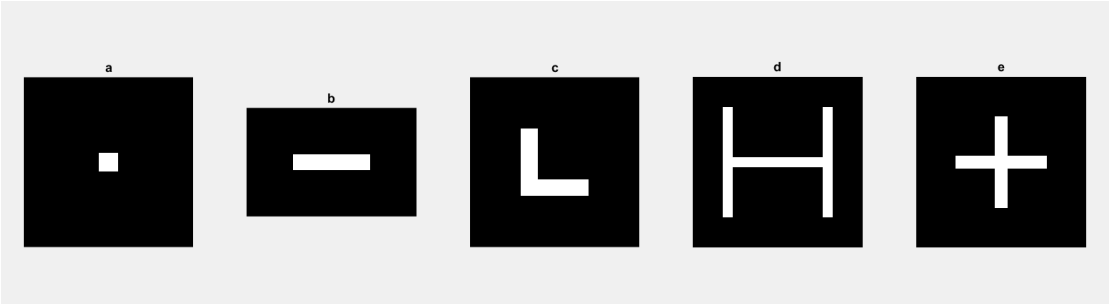
使用區域填充函數填充該區域。

(d)

邊界影像包含一個填充的區域。

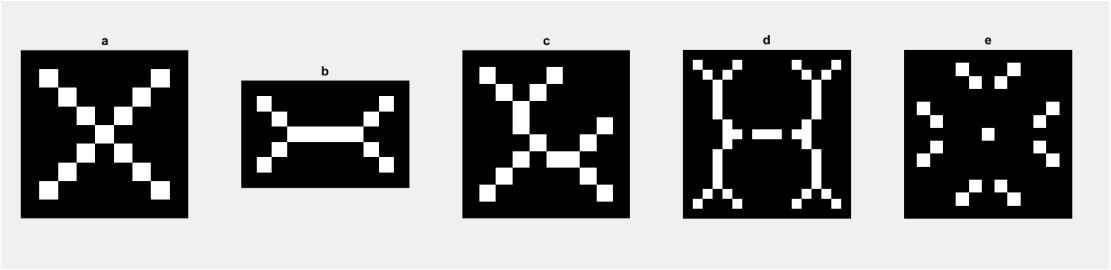
7. (討論)

```
64 —    ak=imskel(a,sq);
65 —    bk=imskel(b,sq);
66 —    ck=imskel(c,sq);
67 —    dk=imskel(d,sq);
68 —    ek=imskel(e,sq);
69 —    subplot(1,5,1),imshow(ak);
70 —    title('a');
71 —    subplot(1,5,2),imshow(bk);
72 —    title('b');
73 —    subplot(1,5,3),imshow(ck);
74 —    title('c');
75 —    subplot(1,5,4),imshow(dk);
76 —    title('d');
77 —    subplot(1,5,5),imshow(ek);
78 —    title('e');
```



8. (討論)

```
ak2=imskel(a,cr);
bk2=imskel(b,cr);
ck2=imskel(c,cr);
dk2=imskel(d,cr);
ek2=imskel(e,cr);
subplot(1,5,1),imshow(ak2);
title('a');
subplot(1,5,2),imshow(bk2);
title('b');
subplot(1,5,3),imshow(ck2);
title('c');
subplot(1,5,4),imshow(dk2);
title('d');
subplot(1,5,5),imshow(ek2);
title('e');
```



10. (實驗)

尋找影像 morph_text.png 上文字中的 “ i “ 上面那一點。

```
m=imread('morph_text.png');
b1=[0 1 1 1 0
    1 1 1 1 1
    1 1 1 1 1
    1 1 1 1 1
    0 1 1 1 0];
b2=ones(7);
b2(2:6,2:6)=~b1;

mb1=imerode(m,b1);
mb2=imerode(~m,b2);
hit_or_miss=mb1&mb2;
[x,y]=find(hit_or_miss==1)
% 找出中心座標

sq=ones(3);
mb=m&~imerode(m,sq);
subplot(1,2,1),imshow(mb);
title('原圖');
mf=imfill(mb,[24,97],sq);
subplot(1,2,2),imshow(mf);
title('i上面那點');
```

```
x =

    68
    24
    24
    68


y =

    42
    58
    97
   153
```

共有 4 個 i，若找第 2 個：