COMPUTER VISION 1

Homework 7

姓名　　：蘇宛琳

系所　　：電信所碩一

學號　　：R05942060

指導教授：傅楸善老師

Computer Vision Report – Homework 7

R05942060 蘇宛琳

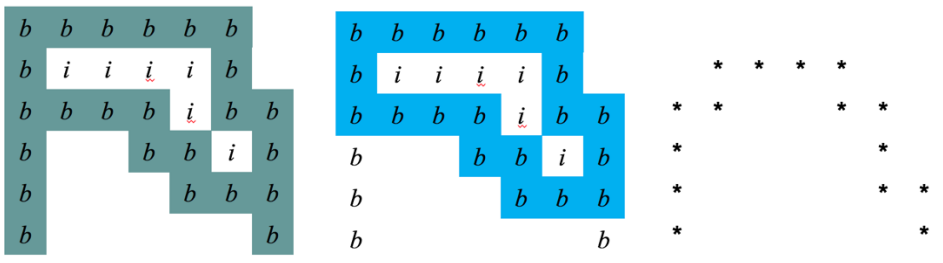
Question :

Write a program to generate thinned image. Down sample lena.bmp from 512*512 to 64*64 first. Sample pixels positions at each 8*8 top-left corner, so everyone will get the same answer .

lena.bmp DownsampleLENA.bmp binarizelena.bmp

The thinning operator is the combination of

1. Mark-Interior/Border-Pixel : Get "*b*" and "*i*"
2. The Pair Relationship : Mark "*b*" pixels that are next to "*i*"
3. Connected Shrink : Shrink marked pixels that are deletable



1.
Get "*b*" and "*i*"

2.
Mark "*b*" pixels that are next to "*i*"

3.
Shrink marked pixels that are deletable

* Thinning function Concept *

Step1. 先將灰階的 512*512 每 8 點取 1 點降取到 64*64，並以 128 為門檻值做二元化。

Step2. 利用Yokoi Connectivity Number function, The Pair Relationship function, Connected Shrink function，產生thinned image。

Source code (Main code)

```
clear;
close;
```

Grayscale LENA image

```
LENA = imread('lena.bmp');
INFO = imfinfo('lena.bmp');
```

Down Sample from 512x512 to 64x64

```
for x = 1 : INFO.Height/8,
    for y = 1 : INFO.Width/8,
        DLENA(x,y) = LENA(x*8-7,y*8-7);
    end;
end;
```

Sample pixels positions at each 8*8 top-left corner

```
imwrite(DLENA,'DownsampleLENA.bmp')
```

Binarize LENA image

```
for x = 1 : INFO.Height/8,
    for y = 1 : INFO.Width/8,
        T = 128;
        if DLENA(x,y) > T,
           NEWLENA(x,y) = 255;
```

```
        else
           NEWLENA(x,y) = 0;
        end;
    end;
end;
imwrite(NEWLENA,'binarizelena.bmp')
```

Thinning function

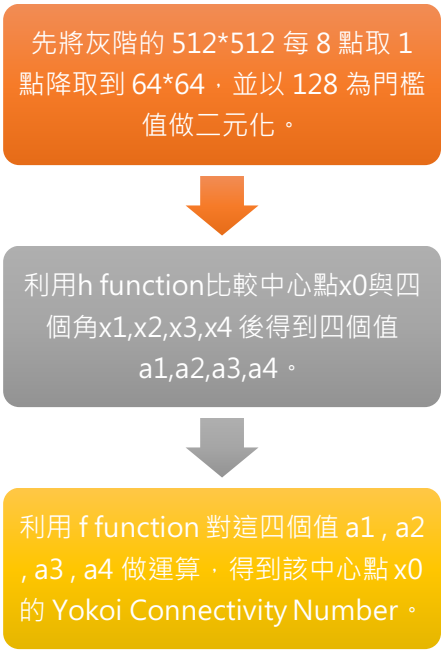
```
A3 = NEWLENA;
change = 1;
while change ~= 0
      A3_pre = A3;
      % Yokoi Connectivity Number
      A1 = YokoiConnectivity(A3);
      % The Pair Relationship
      A2 = PairRelationship(A1);
      % Connected Shrink
      A3 = ConnectedShrink(A3,A2);
      change = sum(abs(A3_pre(:)-A3(:)));
end
figure;
subplot(1,3,1); imshow(DLENA);   title('Downsampling');
subplot(1,3,2); imshow(NEWLENA); title('Binarize');
subplot(1,3,3); imshow(A3);      title('Thinning');
saveas(gcf,'Thinning.jpg');
```

Print Result Label & Write Result to txt file

```
Iycn = YokoiConnectivity(NEWLENA);
Iycn(A3 == 1) = '*';
fid = fopen('Thinning.txt','w');
for k = 1 : size(Iycn,1)
    fprintf(fid,'%c',Iycn(k,:));
    fprintf(fid,'\r\n');
end
fclose(fid);
```

Mark-interior/Border-pixel

* Yokoi Connectivity function Concept*



Connectivity number (0~5)把 border 的情況分類成 6 大類：

0	Isolated	(周圍都屬背景，沒有鄰居)
1	Edge	(周圍1個鄰居)
2	Connecting	(周圍2個鄰居，恰在線上)
3	Branching	(周圍3個鄰居)
4	Crossing	(周圍4個鄰居)
5	Interior	

```
function output = YokoiConnectivity(NEWLENA)

Ib = zeros(size(NEWLENA,1)+2,size(NEWLENA,1)+2);

Ib(2:end-1,2:end-1) = NEWLENA;

[r,c] = find(Ib);

output = char(size(NEWLENA));

for i = 1 : length(r)

    mask = Ib(r(i)-1:r(i)+1,c(i)-1:c(i)+1);

    a = zeros(1,4);

    a(1) = h(mask(5),mask(8),mask(7),mask(4)); % x0,x1,x6,x2

    a(2) = h(mask(5),mask(4),mask(1),mask(2)); % x0,x2,x7,x3

    a(3) = h(mask(5),mask(2),mask(3),mask(6)); % x0,x3,x8,x4

    a(4) = h(mask(5),mask(6),mask(9),mask(8)); % x0,x4,x5,x1

    output(r(i)-1,c(i)-1) = f(a);

end

end
```

* h function Concept*

$$h(x_0,c,d,e)=\begin{cases} q & x_0=c;\ x_0\neq d \text{ or } x_0\neq e \quad (x_0\text{是在邊界}) \\ r & x_0=c=d=e \quad (x_0\text{是在內部}) \\ s & x_0\neq c \quad (\text{沒甚麼意義}) \end{cases}$$

```
function output = h(b,c,d,e)
if b == c,
   if d == b && e == b,
      output = 'r';
   else
      output = 'q';
   end
else
   output = 's';
end
end
```

* f function Concept*

$$f(a_1,a_2,a_3,a_4)=\begin{cases} 5 & \text{if } a_1=a_2=a_3=a_4=r \\ n & n=\#\{a_k \mid a_k=q\} \quad (\text{如此便知道border的種類}) \end{cases}$$

```
function n = f(a)
if all(a == 'r')
   n = num2str(5);
else
   n=num2str(sum(a == 'q'));
end
end
```

input : original symbolic image

output : interior/border image

Pair relationship operator

* Pair Relationship function Concept*

4-connected 中鄰居們有一個以上的鄰居為前景而且本身也為前景，那麼將自己視為一個可以被刪除的點;若本身為孤立點沒有鄰居則不能被刪除。

```
function output = PairRelationship(A1)



Ib = zeros(size(A1,1)+2,size(A1,1)+2);

Ib(2:end-1,2:end-1) = A1;

[r,c] = find(Ib);

output = char(zeros(size(A1)));

for i = 1 : length(r)

    mask = Ib(r(i)-1:r(i)+1,c(i)-1:c(i)+1);

    a = zeros(1,4);

    a(1) = h1(mask(8)); % h1(x1,1)

    a(2) = h1(mask(4)); % h1(x2,1)

    a(3) = h1(mask(2)); % h1(x3,1)

    a(4) = h1(mask(6)); % h1(x4,1)

    output(r(i)-1,c(i)-1) = f1(a,mask(5));

end

end
```

* h1 function Concept*

$$h(a,1)=\begin{cases}1 & \text{if } a=1\\ 0 & \text{otherwise}\end{cases}$$

```
function output = h1(b)


if b == '1'
   output = 1;
else
   output = 0;
end
end
```

* f1 function Concept*

$$y=\begin{cases}q & \text{if } \sum_{n=1}^{4} h(x_n,1)<1 \text{ or } x_0\neq 1 \quad (\text{不能夠被delete})\\ p & \text{if } \sum_{n=1}^{4} h(x_n,1)\geq 1 \text{ or } x_0=1 \quad (\text{可以被delete})\end{cases}$$

```
function y = f1(a,b)


if sum(a) >= 1 && b == '1'
   y = 'p';
else
   y = 'q'; % x0 have no neighbor
end
end
```

input: interior/border image

output: marked image

Connected Shrink

* Connected Shrink function Concept*

前景區域縮小但不使連結斷開 (recursive)

```
function output = ConnectedShrink(NEWLENA,A2)


Ib = zeros(size(NEWLENA,1)+2,size(NEWLENA,1)+2);

Ib(2:end-1,2:end-1) = NEWLENA;

[r,c] = find(A2 == 'p');

r = r + 1;

c = c + 1;


for i = 1 : length(r)

    mask = Ib(r(i)-1:r(i)+1,c(i)-1:c(i)+1);

    a = zeros(1,4);

    a(1) = h2(mask(5),mask(8),mask(7),mask(4)); % x0,x1,x6,x2

    a(2) = h2(mask(5),mask(4),mask(1),mask(2)); % x0,x2,x7,x3

    a(3) = h2(mask(5),mask(2),mask(3),mask(6)); % x0,x3,x8,x4

    a(4) = h2(mask(5),mask(6),mask(9),mask(8)); % x0,x4,x5,x1

    Ib(r(i),c(i)) = f2(a,mask(5));

end

output = Ib(2:end-1,2:end-1);

end
```

* h2 function Concept*

$$h(b,c,d,e)=\begin{cases}1 & \text{if } b=c \text{ and } (b\neq d \text{ or } b\neq e)\\ 0 & \text{otherwise}\end{cases}$$

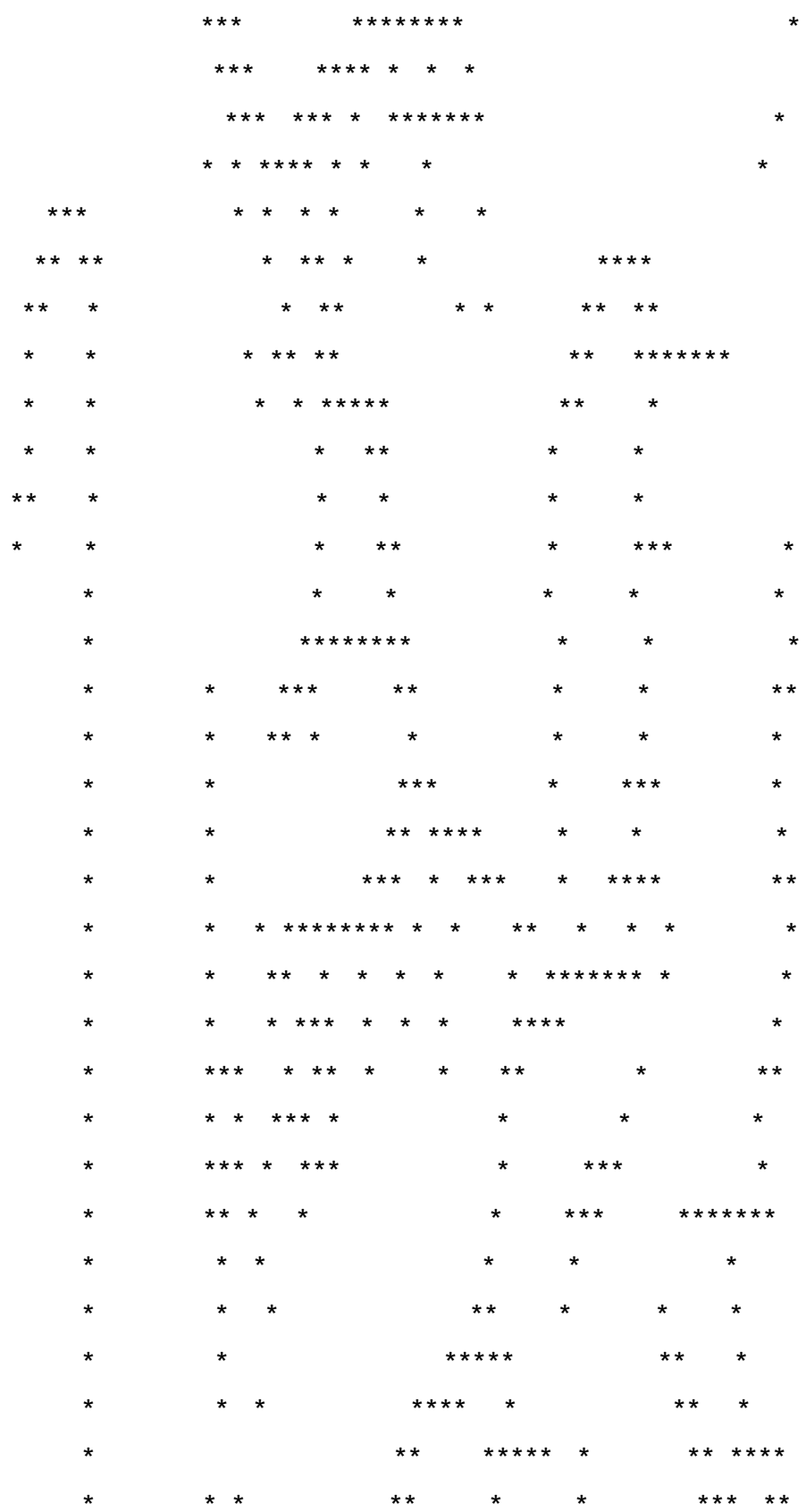
```
function output = h2(b,c,d,e)



if b == c && (d ~= b || e ~= b)

   output = 1;

else

   output = 0;

end

end
```

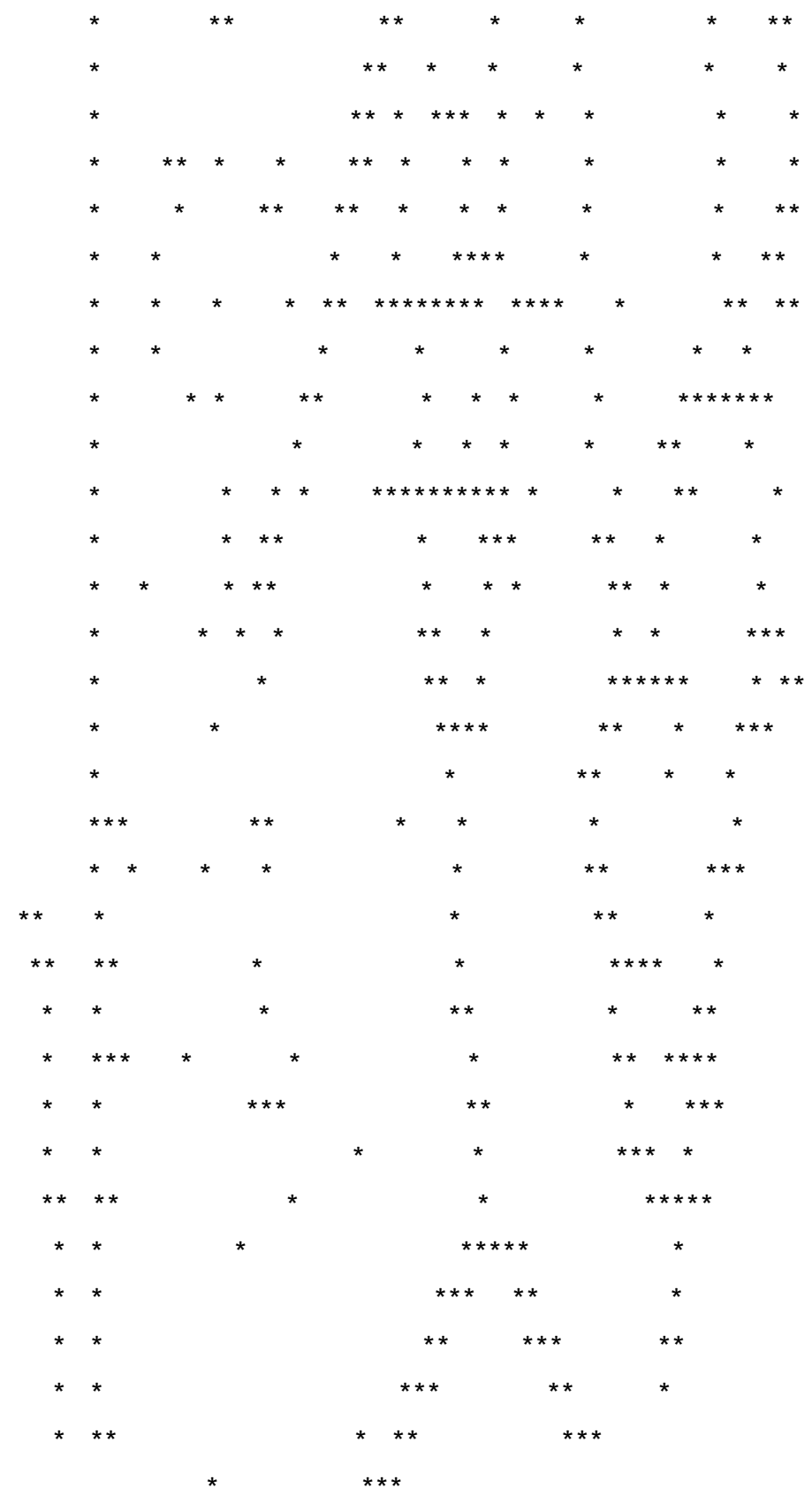
* f2 function Concept*

$$f(a_1,a_2,a_3,a_4,x)=\begin{cases}g & \text{if exactly one of } a_1,a_2,a_3,a_4=1\\ x & \text{otherwise}\end{cases}$$

```
function y = f2(a,x)



if sum(a) == 1

   y = 0;

else

   y = x;

end

end
```

*** 直接貼過來的結果數據 ***





*** 整理過後得到的結果圖 ***

相較於 Connectivity shrink operator，Thinning operator 是一種較對稱的縮小法，縮小的幅度幾乎是上下、左右的對稱。