R05631027_楊皓文_影像處理_作業#3

編譯環境 ：Windows 10 \ Matlab 2016

Part 1: (25%)

R05631027 楊皓文 image processing #3

330 (a) 具有2個梯形函數及1個三角函數

$$\Rightarrow \mu(z) = \begin{cases} 0.5 - \frac{(a-c)-z}{a-c}, & 0 \le z \le a-c \\ 0.5, & a-c \le z \le a-b \\ 1 - \frac{(a-z)}{b}, & a-b \le z \le a \\ 1 - \frac{(z-a)}{b}, & a \le z \le a+b \\ 0.5, & a+b \le z \le a+c \\ 0.5 - \frac{z-(a+c)}{a-c}, & a+c \le z \le 1 \end{cases}$$

(b) 具有2個梯形函數

$$\Rightarrow \mu(z) = \begin{cases} 0.5 - \frac{a+c-z}{c}, & a \le z \le a+c \\ 0.5, & a+c \le z \le b \\ 1 - \frac{b+c-z}{c}, & b \le z \le b+c \\ 1, & b+c \le z \le d \\ 0, & \text{otherwise} \end{cases}$$

(c) 具有2個三角函數

$$\Rightarrow \mu(z) = \begin{cases} 1 - \frac{a-b-z}{a-b}, & 0 \le z \le a-b \\ 1 - \frac{z-(a-b)}{b}, & a-b \le z \le a \\ 1 - \frac{a+b-z}{b}, & a \le z \le a+b \\ 1 - \frac{z-1}{1-(a+b)}, & a+b \le z \le 1 \end{cases}$$

※

No.
Date : :

4.8 $F(u) = \sum_{x=0}^{M-1} f(x) e^{-j2\pi ux/M}$

$f(x) = \frac{1}{M} \sum_{u=0}^{M-1} F(u) e^{j2\pi ux/M}$

(a) $F(u) = \sum_{x=0}^{M-1} \left(\frac{1}{M} \sum_{u=0}^{M-1} F(u) e^{j2\pi ux/M} \right) e^{-j2\pi ux/M}$

$= \frac{1}{M} \sum_{x=0}^{M-1} \sum_{u=0}^{M-1} F(u) e^{j2\pi ux/M} e^{-j2\pi ux/M}$

$= \frac{1}{M} \times M \times F(u) = F(u)$

$f(x) = \frac{1}{M} \sum_{u=0}^{M-1} \left(\sum_{x=0}^{M-1} f(x) e^{-j2\pi ux/M} \right) e^{-j2\pi ux/M}$

$= \frac{1}{M} \sum_{u=0}^{M-1} \sum_{x=0}^{M-1} f(x) e^{-j2\pi ux/M} e^{-j2\pi ux/M}$

$= f(x)$

(b) $F(u) = \sum_{x=0}^{M-1} \left[\frac{1}{M} \sum_{r=0}^{M-1} F(r) e^{-j2\pi rx/M} \right] e^{-j2\pi ux/M}$

$= \frac{1}{M} \sum_{r=0}^{M-1} F(r) \left[\sum_{x=0}^{M-1} e^{-j2\pi rx/M} \cdot e^{-j2\pi ux/M} \right]$

$= F(u)$ ※

4.12

period = 2 mm

$\Rightarrow$ max fregency $= \frac{1}{\text{period}} = 0.5$ cycles/mm

∴ 為避免 aliasing，至少須高於 max fregency 的兩倍的 sampling rate

$\Rightarrow 2 \cdot 0.5 = 1$ sample/mm ※

4.16

(a) $F[f(x,y)\, e^{j2\pi(u_0x+v_0y)}]$

$$= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[f(x,y)\, e^{j2\pi(u_0x+v_0y)}]\, e^{-j2\pi(ux/M+vy/N)}$$

$$= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\, e^{-j2\pi[(u-u_0)x/M+(v-v_0)y/N]}$$

$$= F(u-u_0, v-v_0)$$

(b) $F^{-1}[F(u,v)\, e^{-j2\pi(ux_0+vy_0)}]$

$$= \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}[F(u,v)\, e^{-j2\pi(ux_0+vy_0)}]$$

$$= \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)\, e^{j2\pi[u(x-x_0)/M+v(y-y_0)/N]}$$

$$= f(x-x_0, y-y_0)$$ ※

4.21

需要進行 padding 的原因最主要是因為要在 Discrete Fourier Transform 的 domain，建立一個 buffer。因此，不管什麼形式的 padding，應該都可達到相同效果。 ※

Part 2: (25%)

程式碼 ：

```
function varargout = HW02(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @HW02_OpeningFcn, ...
                   'gui_OutputFcn',  @HW02_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end



function HW02_OpeningFcn(hObject, eventdata, handles, varargin)

handles.w = 0;
w = num2str(handles.w);
set(handles.edit1,'string', w);
set(handles.edit1, 'Max', 9);

handles.output = hObject;
guidata(hObject, handles);



function varargout = HW02_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;



function pushbutton1_Callback(hObject, eventdata, handles)

filename = uigetfile({'*.jpg;*.tif;*.png;*.gif;*.bmp','All Image Files'});
```

```
handles.I = imread(filename);

[handles.row, handles.col, handles.lay] = size(handles.I);

if handles.lay == 1
	handles.I_gray = handles.I;
	fprintf('gray image\n');
elseif handles.lay == 3
	handles.R = uint8(handles.I(:, :, 1));
	handles.G = uint8(handles.I(:, :, 2));
	handles.B = uint8(handles.I(:, :, 3));
	handles.I_gray = 0.299*handles.R + 0.587*handles.G + 0.114*handles.B;
	fprintf('color image\n');
end

axes(handles.axes1);
imshow(handles.I_gray);

handles.output = hObject;
guidata(hObject, handles);



function radiobutton1_Callback(hObject, eventdata, handles)

set(handles.radiobutton1, 'Value', 1);
set(handles.radiobutton2, 'Value', 0);

w(hObject, eventdata, handles);



function radiobutton2_Callback(hObject, eventdata, handles)

set(handles.radiobutton1, 'Value', 0);
set(handles.radiobutton2, 'Value', 1);

w(hObject, eventdata, handles);



function radiobutton4_Callback(hObject, eventdata, handles)
```

```
set(handles.radiobutton4, 'Value', 1);
set(handles.radiobutton5, 'Value', 0);
set(handles.radiobutton6, 'Value', 0);

w(hObject, eventdata, handles);



function radiobutton5_Callback(hObject, eventdata, handles)

set(handles.radiobutton4, 'Value', 0);
set(handles.radiobutton5, 'Value', 1);
set(handles.radiobutton6, 'Value', 0);

w(hObject, eventdata, handles);



function radiobutton6_Callback(hObject, eventdata, handles)

set(handles.radiobutton4, 'Value', 0);
set(handles.radiobutton5, 'Value', 0);
set(handles.radiobutton6, 'Value', 1);

w(hObject, eventdata, handles);


function w(hObject, eventdata, handles)
% Laplacian 3x3
if handles.radiobutton1.Value == 1 && handles.radiobutton4.Value == 1
	handles.w = [0 1 0
                     1 -4 1
                     0 1 0];
% Sobel_x 3x3
elseif handles.radiobutton1.Value == 1 && handles.radiobutton5.Value == 1
	handles.w = [1 0 -1
                     2 0 -2
                     1 0 -1];
% Smoothing 3x3
elseif handles.radiobutton1.Value == 1 && handles.radiobutton6.Value == 1
	handles.w = [1 1 1
```

```
                        1 1 1
                        1 1 1];
% Laplacian 5x5
elseif handles.radiobutton2.Value == 1 && handles.radiobutton4.Value == 1
    handles.w = [0 0 1 0 0
                        0 1 2 1 0
                        1 2 -16 2 1
                        0 1 2 1 0
                        0 0 1 0 0];
% Sobel_x 5x5
elseif handles.radiobutton2.Value == 1 && handles.radiobutton5.Value == 1
    handles.w = [1 2 0 -2 -1
                        4 8 0 -8 -4
                        6 12 0 -12 -6
                        4 8 0 -8 -4
                        1 2 0 -2 -1];
% Smoothing 5x5
elseif handles.radiobutton2.Value == 1 && handles.radiobutton6.Value == 1
    handles.w = [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];

end
w = num2str(handles.w);
set(handles.edit1,'String', w);

handles.output = hObject;
guidata(hObject, handles);



function edit1_Callback(hObject, eventdata, handles)



function edit1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

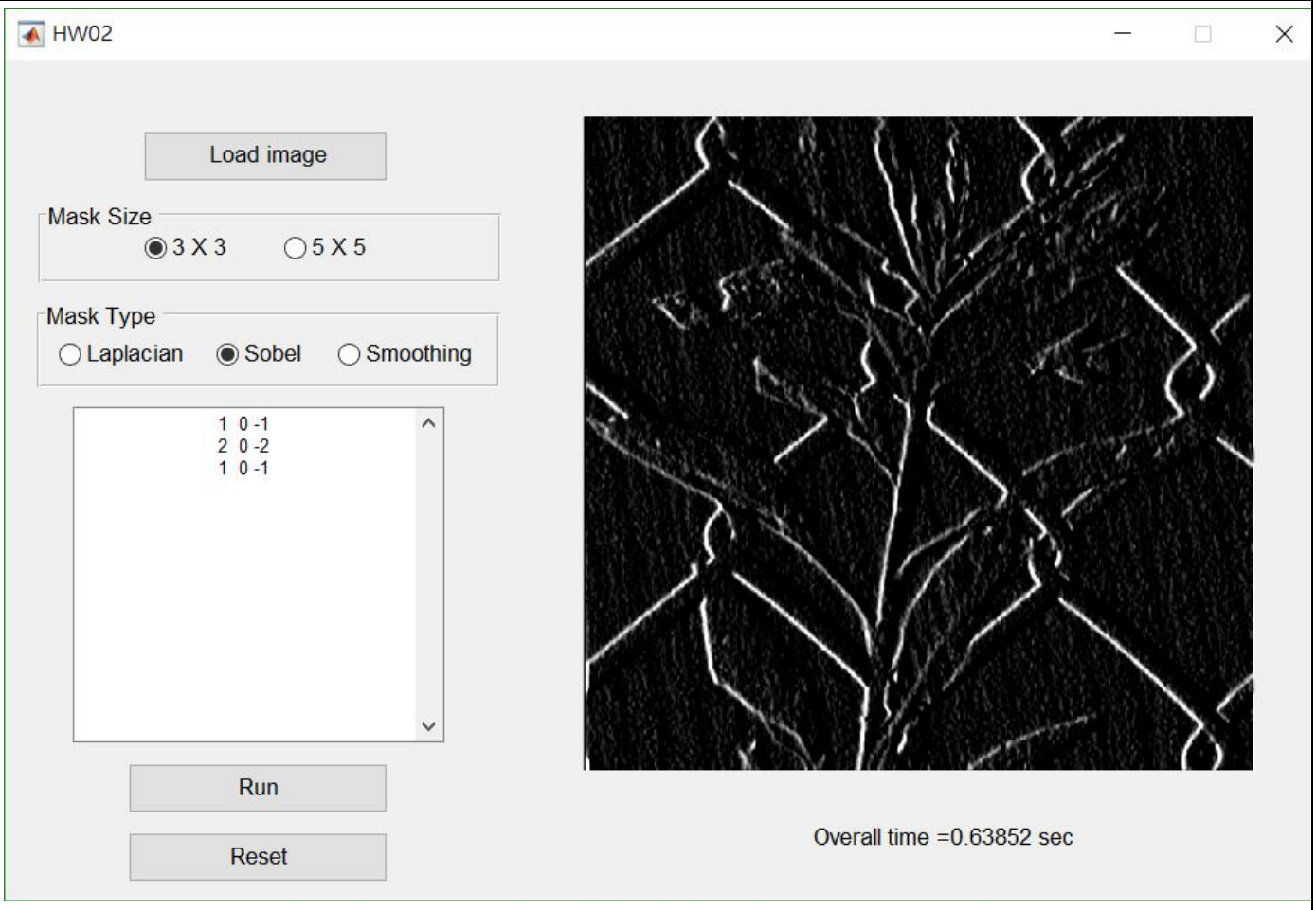
```
function pushbutton2_Callback(hObject, eventdata, handles)

handles.w = get(handles.edit1, 'String');
handles.w = str2num(handles.w);

% time start
timer1=tic;

% p 隨 mask 的大小改變，例如: 3x3 => p = 1, 5x5 => p = 2
p = ( size(handles.w, 1) - 1 ) / 2;

% smoothing  的分母
N = sum(sum(handles.w));
if N == 0
	N = 1;
end

I3 = double(zeros(handles.row+2*p, handles.col+2*p));
I3(1 + p:handles.row + p, 1 + p:handles.col + p) = handles.I_gray;
I4 = double(zeros(handles.row, handles.col));

for x = 1+p:handles.row+p
	for y = 1+p:handles.col+p
		a = 1;
		for m = x-p:x+p
			b = 1;
			for n = y-p:y+p
				%  將 mask 裡面的數全部加起來
				I4(x-p, y-p) = I4(x-p, y-p) + round( handles.w(a, b)*I3(m, n)/N );
				b = b+1;
			end
			a = a+1;
		end
	end
end
I4 = uint8(I4);
axes(handles.axes1); imshow(I4);

% time end
```

```
set(handles.text2, 'String', ['Overall time =', num2str( toc(timer1) ), ' sec'] );

handles.output = hObject;
guidata(hObject, handles);



function pushbutton3_Callback(hObject, eventdata, handles)

set(handles.radiobutton1, 'Value', 0);
set(handles.radiobutton2, 'Value', 0);
set(handles.radiobutton4, 'Value', 0);
set(handles.radiobutton5, 'Value', 0);
set(handles.radiobutton6, 'Value', 0);
handles.w = 0;
w(hObject, eventdata, handles);
set(handles.text2, 'String', 'Overall time :' );

axes(handles.axes1);
imshow(handles.I_gray);
```

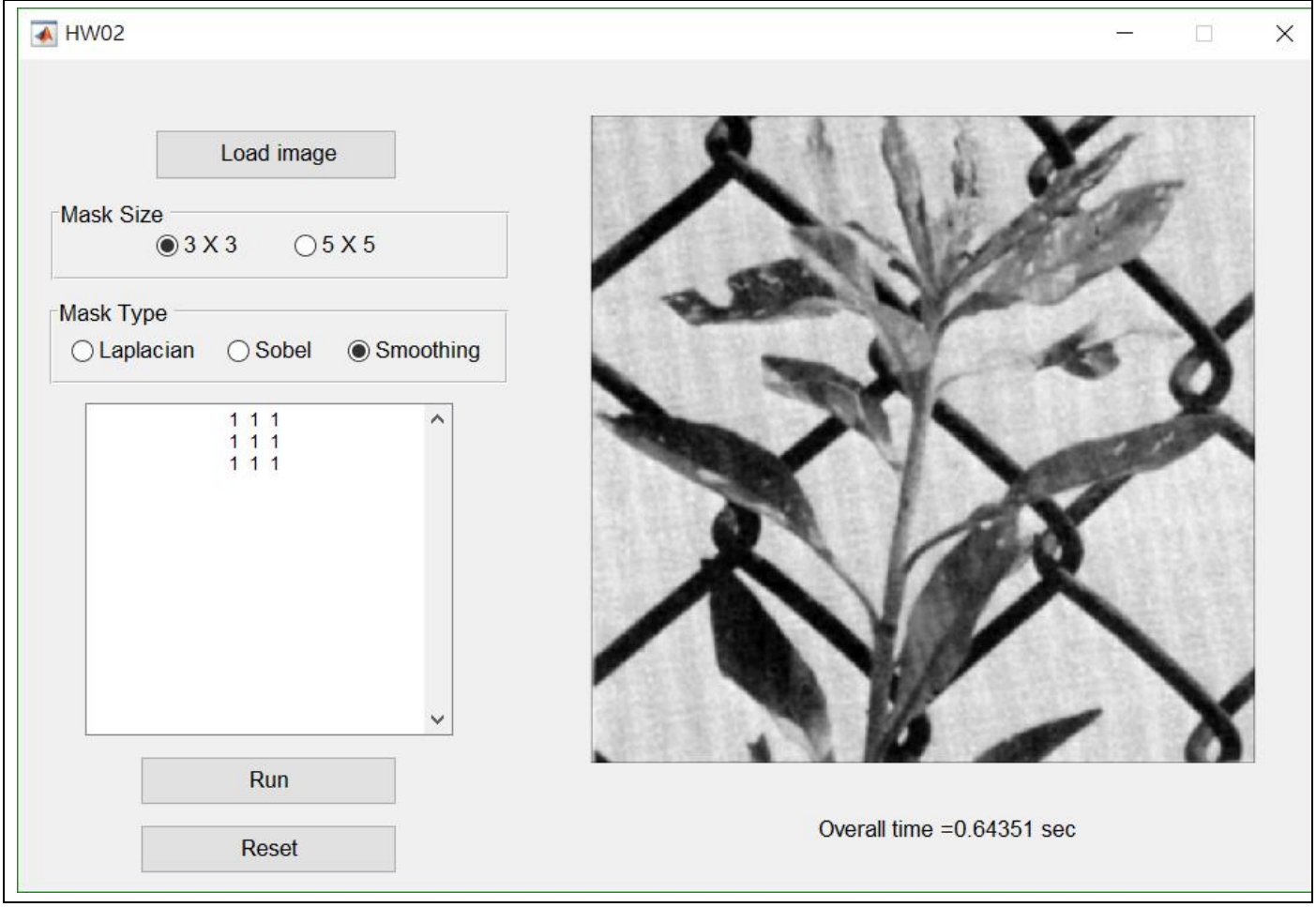
結果呈現 ：

Laplacian 3x3

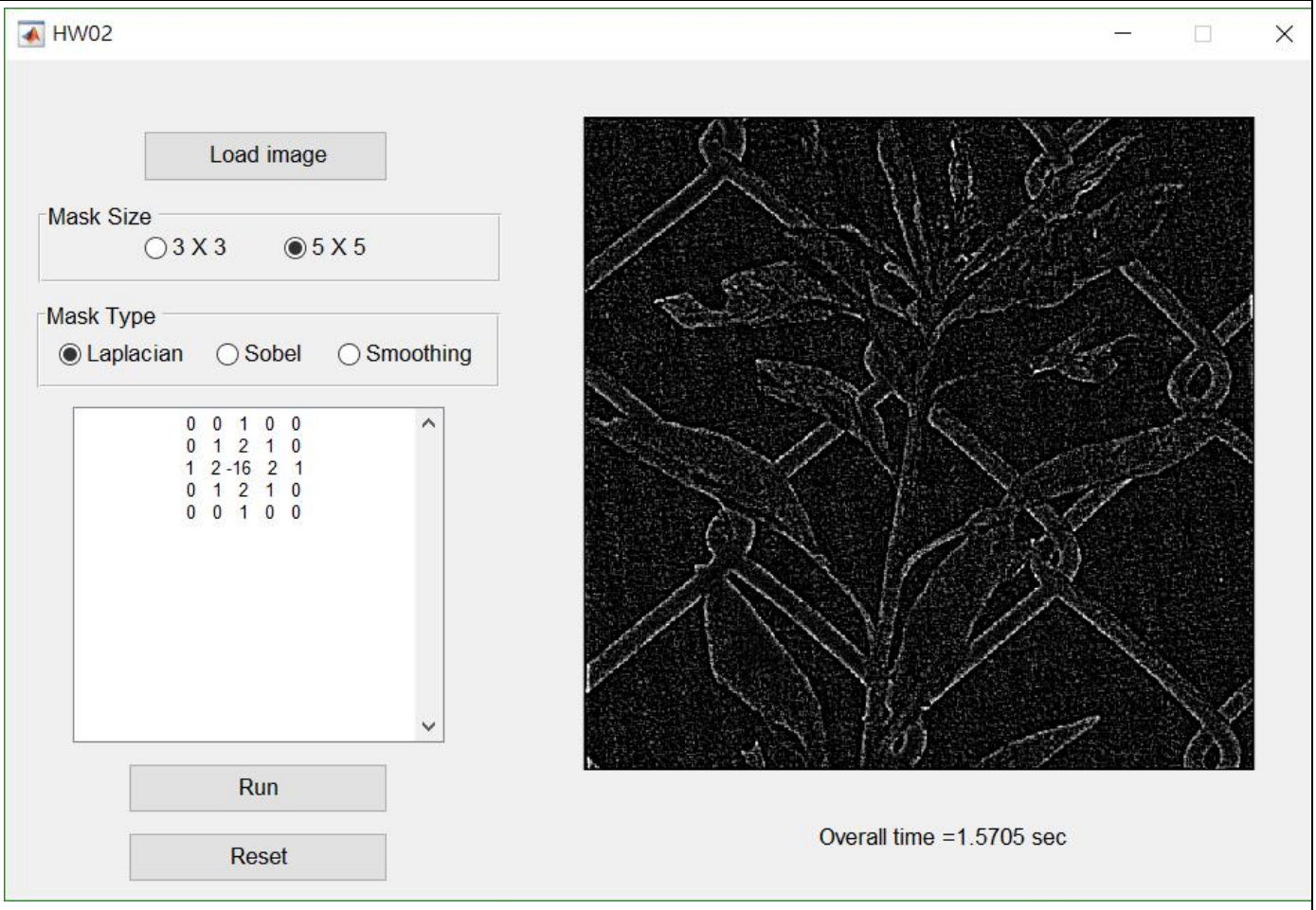
Sobel_x 3x3



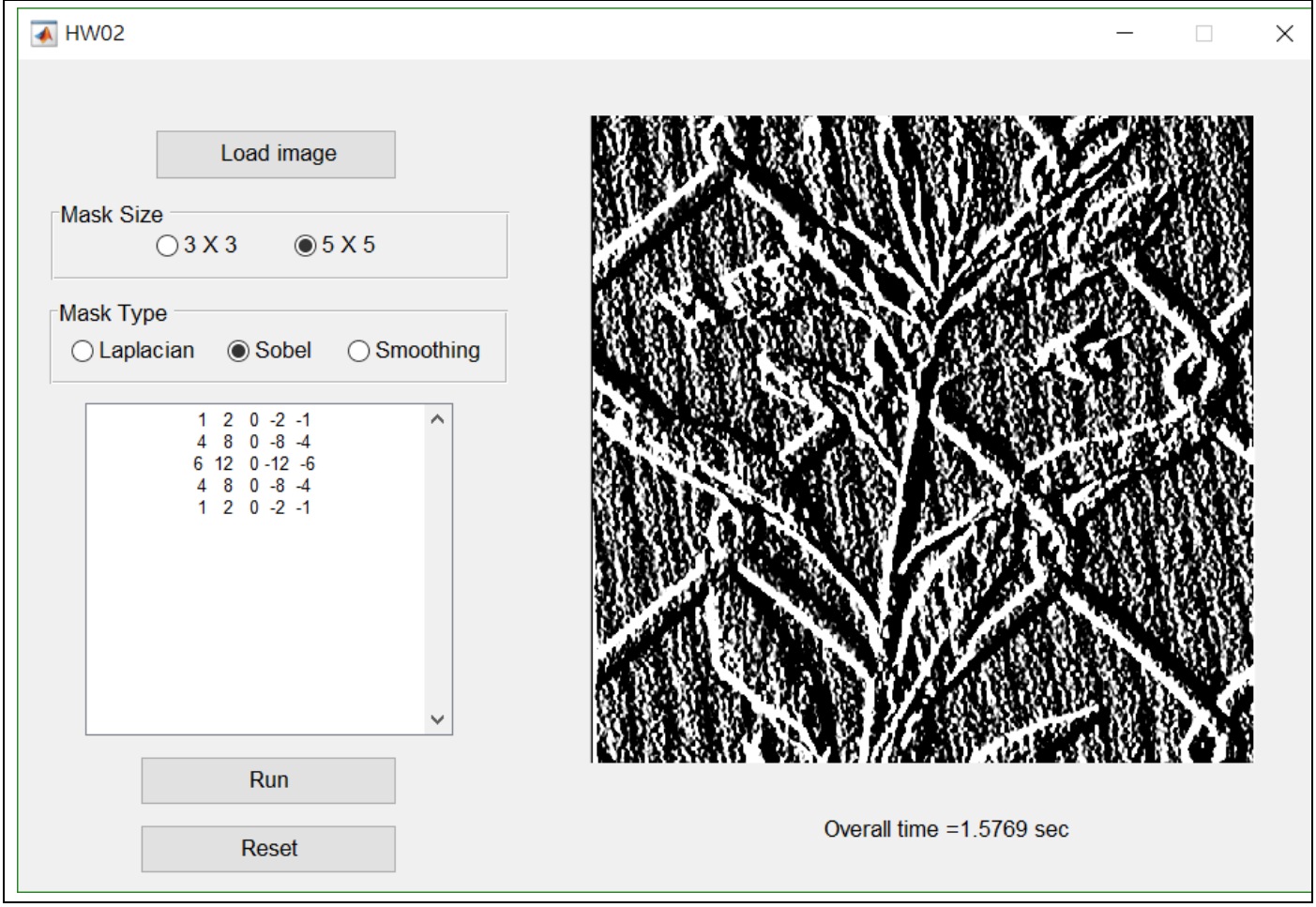
Smoothing 3x3



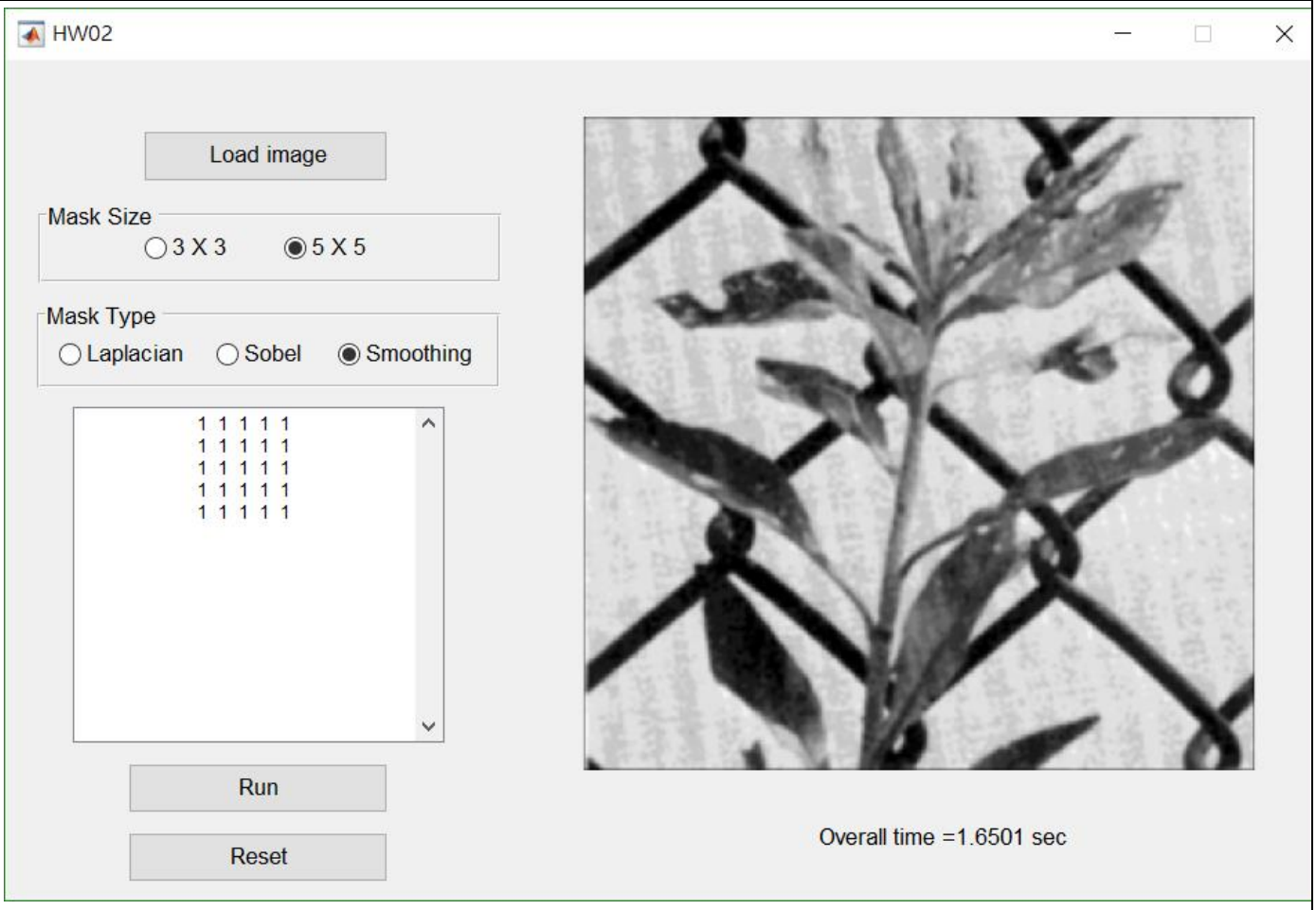
Laplacian 5x5



Sobel_x 5x5



Smoothing 5x5



結果討論 ：

Mask 的大小對於輸出的影像會有很大的影響，像是從 smoothing 就可以看出 5x5 較 3x3 模糊，而 Laplacian 和 Sobel 因為 5x5 沒有明確的定義，所以 weight 不太一樣，以至於輸出的影像也是大不相同。

至於計算時間的部分，我設計在 GUI 的右下角，可以看到 mask 的大小很明顯地影響計算速度，只要是 5x5 計算時間都大於 3x3，而不同 type 的 mask 大小相同的 mask，計算時間則差不多。

Part 3: (25%)

程式碼 ：

```
function varargout = HW03(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',          mfilename, ...
                   'gui_Singleton',    gui_Singleton, ...
                   'gui_OpeningFcn', @HW03_OpeningFcn, ...
                   'gui_OutputFcn',    @HW03_OutputFcn, ...
                   'gui_LayoutFcn',    [] , ...
                   'gui_Callback',     []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
```

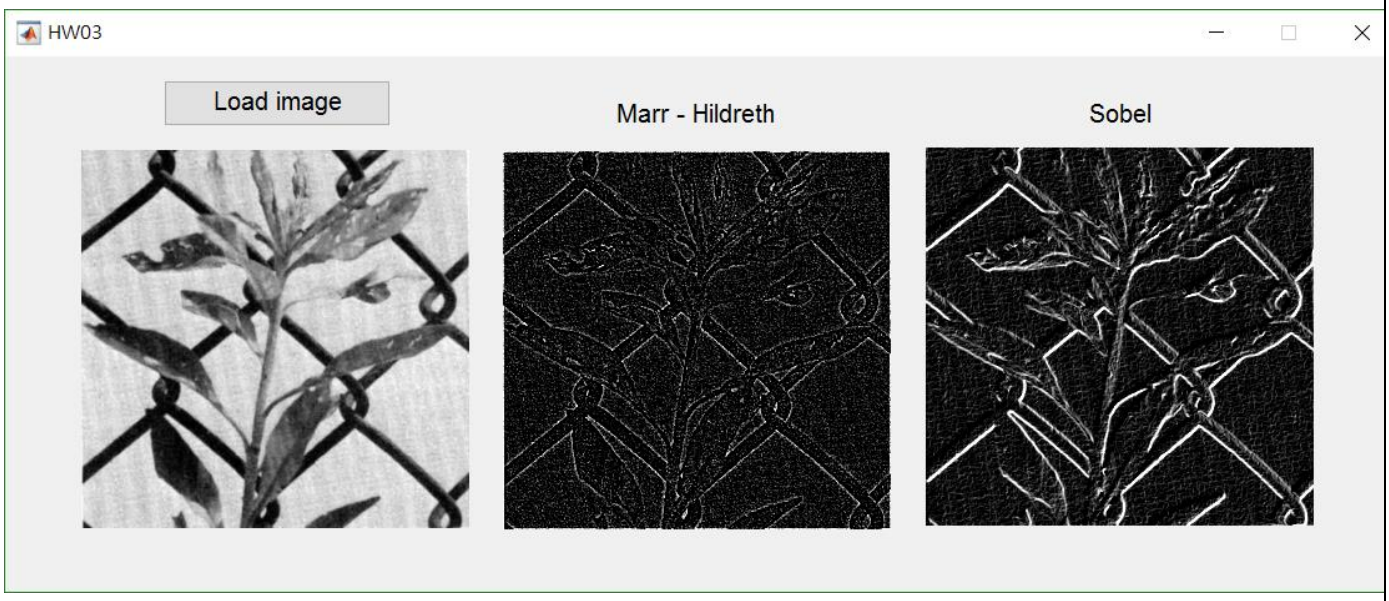
```
if nargout
	[varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
	gui_mainfcn(gui_State, varargin{:});
end



function HW03_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;
guidata(hObject, handles);



function varargout = HW03_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;



function pushbutton1_Callback(hObject, eventdata, handles)

% 原圖
filename = uigetfile({'*.jpg;*.tif;*.png;*.gif;*.bmp','All Image Files'});
handles.I = imread(filename);

[handles.row, handles.col, handles.lay] = size(handles.I);

if handles.lay == 1
	handles.I_gray = handles.I;
	fprintf('gray image\n');
elseif handles.lay == 3
	handles.R = uint8(handles.I(:, :, 1));
	handles.G = uint8(handles.I(:, :, 2));
	handles.B = uint8(handles.I(:, :, 3));
	handles.I_gray = 0.299*handles.R + 0.587*handles.G + 0.114*handles.B;
	fprintf('color image\n');
end

axes(handles.axes1);
```

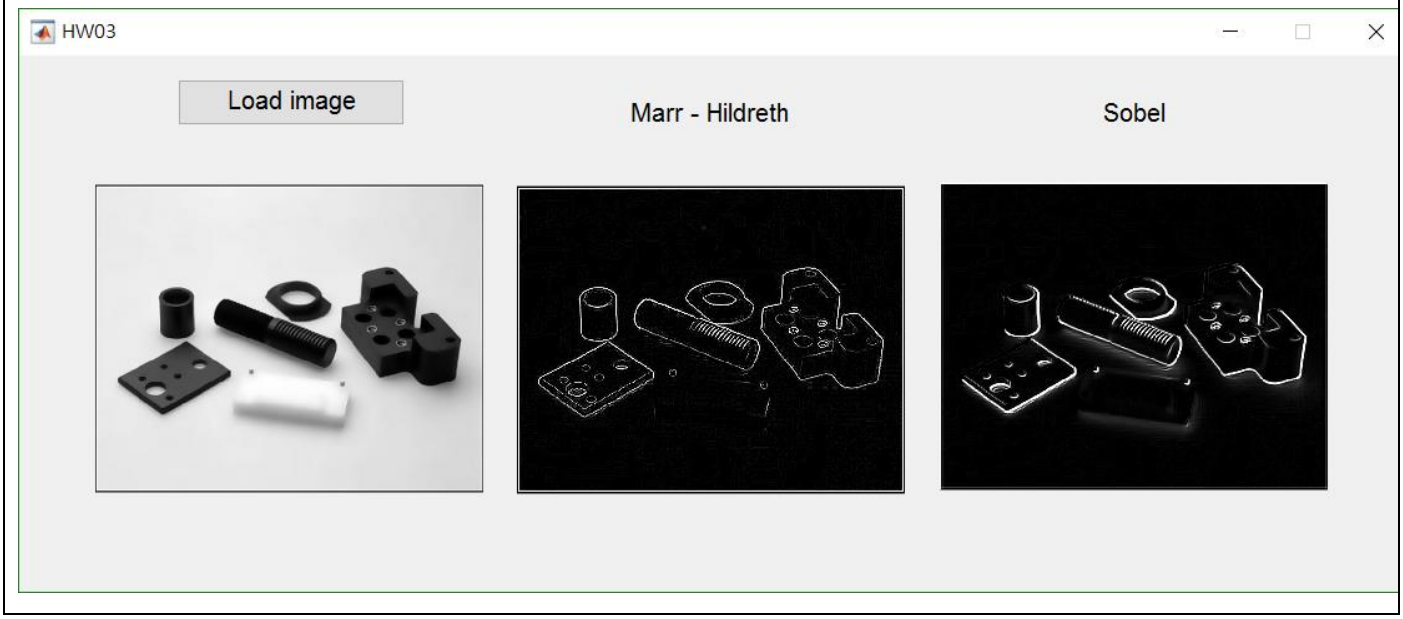
```
imshow(handles.I_gray);

% Marr-Hildreth
handles.LoG = [0 0 -1 0 0
               0 -1 -2 -1 0
               -1 -2 16 -2 -1
               0 -1 -2 -1 0
               0 0 -1 0 0];

% p 隨 mask 的大小改變，例如: 3x3 => p = 1, 5x5 => p = 2
p = ( size(handles.LoG, 1) - 1 ) / 2;

I3 = double(zeros(handles.row+2*p, handles.col+2*p));
I3(1 + p:handles.row + p, 1 + p:handles.col + p) = handles.I_gray;
I4 = double(zeros(handles.row, handles.col));

for x = 1+p:handles.row+p
    for y = 1+p:handles.col+p
        a = 1;
        for m = x-p:x+p
            b = 1;
            for n = y-p:y+p
%                 將 mask 裡面的數全部加起來
                I4(x-p, y-p) = I4(x-p, y-p) + round( handles.LoG(a, b)*I3(m, n) );
                b = b+1;
            end
            a = a+1;
        end
    end
end
I4 = uint8(I4);
axes(handles.axes2); imshow(I4);

% Sobel_x
handles.Sx =  [-1 0 1
               -2 0 2
               -1 0 1];

% p 隨 mask 的大小改變，例如: 3x3 => p = 1, 5x5 => p = 2
p = ( size(handles.Sx, 1) - 1 ) / 2;

I3 = double(zeros(handles.row+2*p, handles.col+2*p));
```

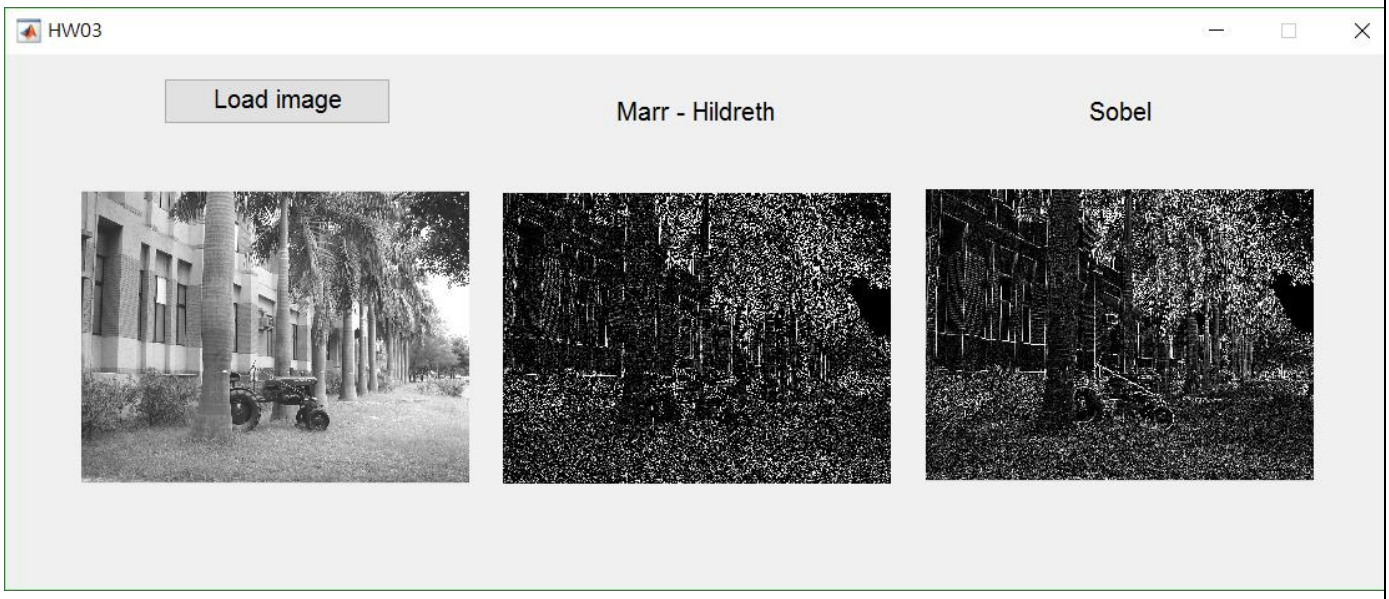
```
I3(1 + p:handles.row + p, 1 + p:handles.col + p) = handles.I_gray;
I4 = double(zeros(handles.row, handles.col));

for x = 1+p:handles.row+p
    for y = 1+p:handles.col+p
        a = 1;
        for m = x-p:x+p
            b = 1;
            for n = y-p:y+p
%                 將 mask 裡面的數全部加起來
                I4(x-p, y-p) = I4(x-p, y-p) + round( handles.Sx(a, b)*I3(m, n) );
                b = b+1;
            end
            a = a+1;
        end
    end
end
I5 = uint8(I4);

% Sobel_y
handles.Sy =    [-1 -2 -1
                 0 0 0
                 1 2 1];
% p 隨 mask 的大小改變，例如: 3x3 => p = 1, 5x5 => p = 2
p = ( size(handles.Sy, 1) - 1 ) / 2;

I3 = double(zeros(handles.row+2*p, handles.col+2*p));
I3(1 + p:handles.row + p, 1 + p:handles.col + p) = handles.I_gray;
I4 = double(zeros(handles.row, handles.col));

for x = 1+p:handles.row+p
    for y = 1+p:handles.col+p
        a = 1;
        for m = x-p:x+p
            b = 1;
            for n = y-p:y+p
%                 將 mask 裡面的數全部加起來
                I4(x-p, y-p) = I4(x-p, y-p) + round( handles.Sy(a, b)*I3(m, n) );
                b = b+1;
            end
            a = a+1;
        end
```

```
    end
end
I6 = uint8(I4);

% Sobel_x + Sobel_y
axes(handles.axes3); imshow(I5+I6);

handles.output = hObject;
guidata(hObject, handles);
```

結果呈現 ：







結果討論 ：

由上面四種不同圖片可以發現，Marr – Hildreth 整體的輪廓較 Sobel 均勻，因為 Sobel 是利用 x 方向和 y 方向的一階導數相加，但是由於微分的方向不同，因此出來的圖片輪廓，總是會有某個方向不太完整；而 Marr – Hildreth 就沒有這個問題，他是使用 Laplacian of Gaussian 的 mask 來對圖片 convolution，而這個 mask 的特點之一就是上下左右對稱，因此運算出來的輪廓在四個方向較均勻。

Part 4: (25%)

程式碼 ：

```
function varargout = HW04(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @HW04_OpeningFcn, ...
```

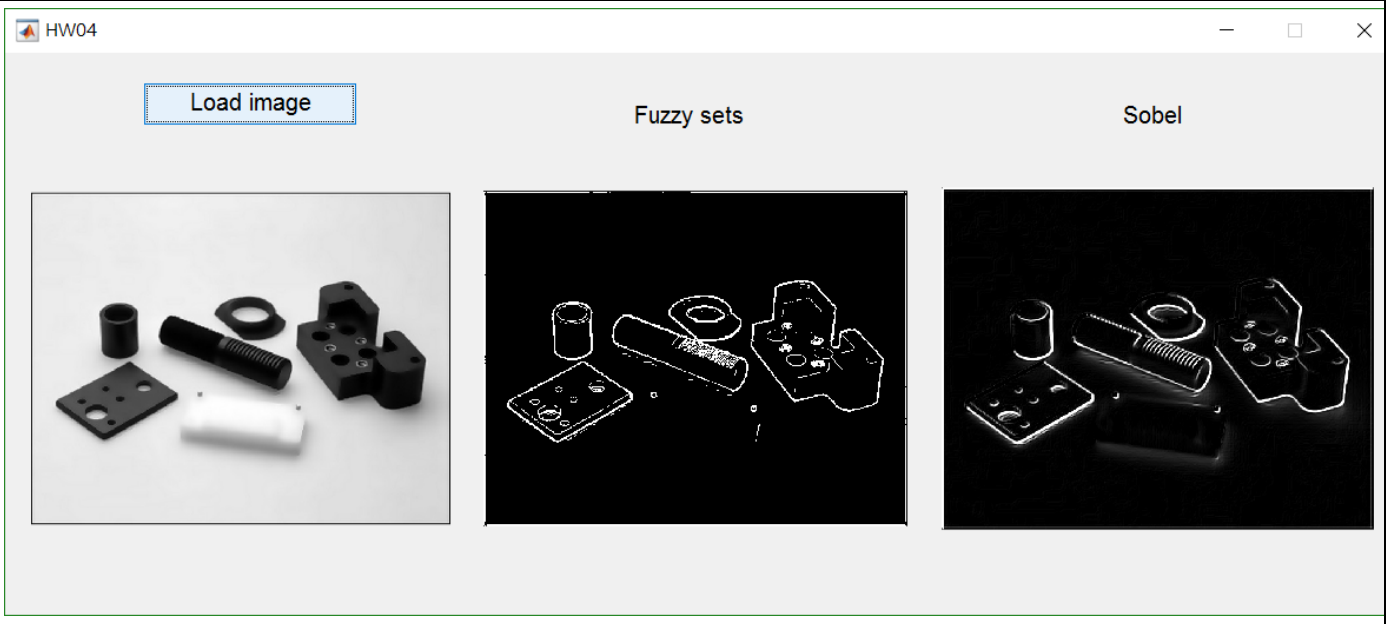
```
                                'gui_OutputFcn',    @HW04_OutputFcn, ...
                                'gui_LayoutFcn',    [] , ...
                                'gui_Callback',     []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end



function HW04_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;
guidata(hObject, handles);



function varargout = HW04_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;



function pushbutton1_Callback(hObject, eventdata, handles)

% 原圖
filename = uigetfile({'*.jpg;*.tif;*.png;*.gif;*.bmp','All Image Files'});
handles.I = imread(filename);

[handles.row, handles.col, handles.lay] = size(handles.I);

if handles.lay == 1
    handles.I_gray = handles.I;
    fprintf('gray image\n');
elseif handles.lay == 3
    handles.R = uint8(handles.I(:, :, 1));
    handles.G = uint8(handles.I(:, :, 2));
```

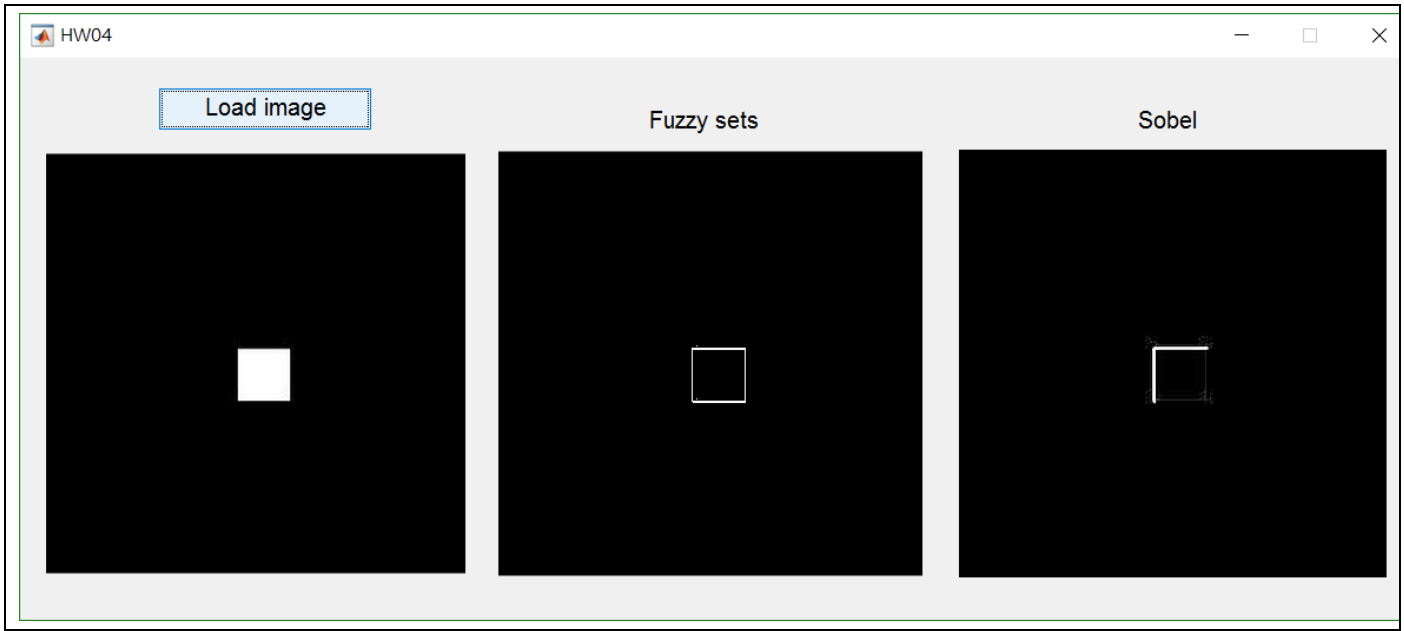
```
    handles.B = uint8(handles.I(:, :, 3));
    handles.I_gray = 0.299*handles.R + 0.587*handles.G + 0.114*handles.B;
    fprintf('color image\n');
end

axes(handles.axes1);
imshow(handles.I_gray);

% Sobel_x
handles.Sx =    [-1 0 1
                 -2 0 2
                 -1 0 1];

% p 隨 mask 的大小改變，例如: 3x3 => p = 1, 5x5 => p = 2
p = ( size(handles.Sx, 1) - 1 ) / 2;

I3 = double(zeros(handles.row+2*p, handles.col+2*p));
I3(1 + p:handles.row + p, 1 + p:handles.col + p) = handles.I_gray;
I4 = double(zeros(handles.row, handles.col));

for x = 1+p:handles.row+p
    for y = 1+p:handles.col+p
        a = 1;
        for m = x-p:x+p
            b = 1;
            for n = y-p:y+p
                % 將 mask 裡面的數全部加起來
                I4(x-p, y-p) = I4(x-p, y-p) + round( handles.Sx(a, b)*I3(m, n) );
                b = b+1;
            end
            a = a+1;
        end
    end
end
I5 = uint8(I4);

% Sobel_y
handles.Sy =    [-1 -2 -1
                 0 0 0
                 1 2 1];
% p 隨 mask 的大小改變，例如: 3x3 => p = 1, 5x5 => p = 2
p = ( size(handles.Sy, 1) - 1 ) / 2;
```

```
I3 = double(zeros(handles.row+2*p, handles.col+2*p));
I3(1 + p:handles.row + p, 1 + p:handles.col + p) = handles.I_gray;
I4 = double(zeros(handles.row, handles.col));

for x = 1+p:handles.row+p
    for y = 1+p:handles.col+p
        a = 1;
        for m = x-p:x+p
            b = 1;
            for n = y-p:y+p
                % 將 mask 裡面的數全部加起來
                I4(x-p, y-p) = I4(x-p, y-p) + round( handles.Sy(a, b)*I3(m, n) );
                b = b+1;
            end
            a = a+1;
        end
    end
end
I6 = uint8(I4);

% Sobel_x + Sobel_y
axes(handles.axes3); imshow(I5+I6);

% Fuzzy sets
I7 = double(zeros(handles.row, handles.col));
for x = 1+p:handles.row+p
    for y = 1+p:handles.col+p
        d = zeros(3);
        a = 1;
        for m = x-p:x+p
            b = 1;
            for n = y-p:y+p
                % intensity difference
                d(a, b) = I3(m, n) - I3(x, y);
                b = b + 1;
            end
            a = a + 1;
        end

        if abs( d(1, 2) ) <= 10 && abs( d(2, 3) ) <= 10
            I7(x-p, y-p) = 0;
```

```
            elseif abs( d(2, 3) ) <= 10 && abs( d(3, 2) ) <= 10
                I7(x-p, y-p) = 0;
            elseif abs( d(3, 2) ) <= 10 && abs( d(2, 1) ) <= 10
                I7(x-p, y-p) = 0;
            elseif abs( d(2, 1) ) <= 10 && abs( d(1, 2) ) <= 10
                I7(x-p, y-p) = 0;
            else
                I7(x-p, y-p) = 255;
            end
        end
end

I7 = uint8(I7);
axes(handles.axes2); imshow(I7);
```

結果呈現 ：





結果討論 ：

相較於 Sobel 總是會有某個方向不太完整，Fuzzy sets 因為可以自己決定上下左右各種組合的 intensity difference 小於正負多少就會變成黑色，像我就設定若上下左右各種組合的 intensity difference 若絕對值小於 10，則該 pixel 就設定為黑色，其他則為白色，因此可以看到較無梯度變化的地方都是黑色，而有明顯梯度變化的地方則為白色。另一方面，由於影像只有黑白，因此輸出的影像輪廓對比又更清楚。