Edge Detection (Turunan ke dua)

Kuliah : Pengolahan Citra

Dosen Eri Prasetyo

Using Second Derivatives for Image Enhancement

- The 2nd derivative is more useful for image enhancement than the 1st derivative
  - Stronger response to fine detail
  - Simpler implementation

- The first sharpening filter we will look at is the *Laplacian*
  - Isotropic
  - One of the simplest sharpening filters
  - We will look at a digital implementation

The Laplacian

- The Laplacian is defined as follows:

$$\nabla^2 f = \frac{\partial^2 f}{\partial^2 x} + \frac{\partial^2 f}{\partial^2 y}$$

- where the partial 1st order derivative in the x direction is defined as follows:

$$\frac{\partial^2 f}{\partial^2 x} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

- and in the y direction as follows:

$$\frac{\partial^2 f}{\partial^2 y} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$

Laplacien Operator (seconde derivative)

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$$f(x, y) = xy^3 + \sin(x) \Longrightarrow \nabla^2 f = -\sin(x) + 6xy$$

/x

/y

$$\nabla f = \begin{bmatrix} y^3 + \cos(x) & 3xy^2 \end{bmatrix}$$

/x

/x

/y

/y

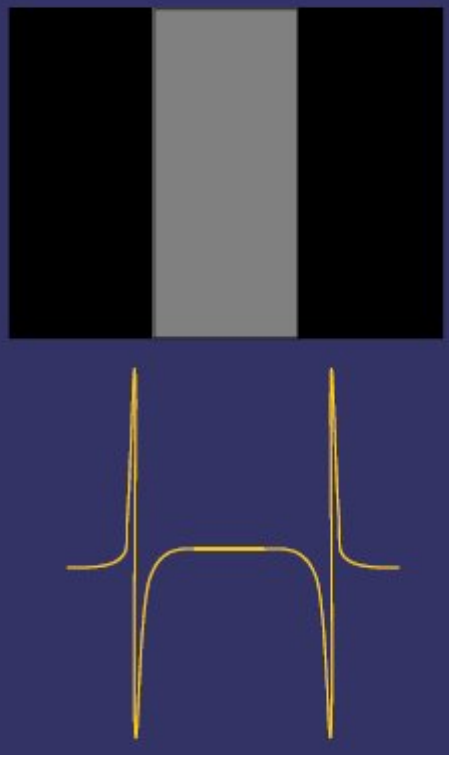
$$\begin{bmatrix} -\sin(x) & 3y^2 \\ 3y^2 & 6xy \end{bmatrix} = Hf$$

Laplacien operator

0	-1	0
-1	4	-1
0	-1	0

- Noise sensitive
- Double ligne generation
- 0 crossing

$$R = \sum w_i f_i = 4f_5 - (f_2 + f_4 + f_6 + f_8)$$



The Laplacian (cont…)

- So, the Laplacian can be given as follows:

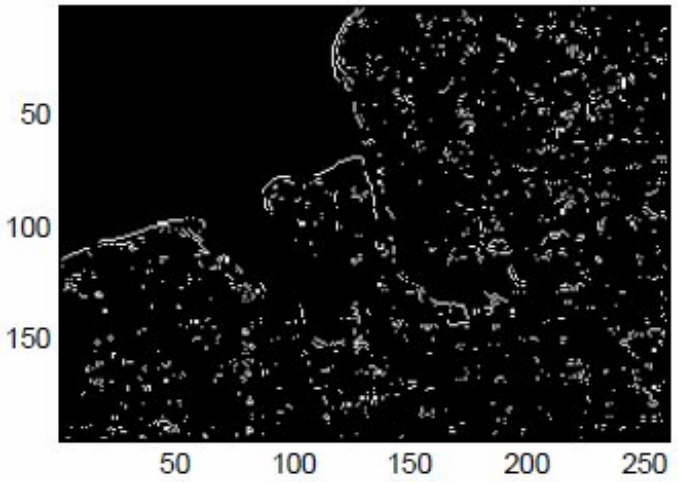
$$\nabla^2 f = [f(x+1, y) f(x-1, y) + f(x, y+1) + f(x, y-1)] - 4 f(x, y)$$

- We can easily build a filter based on this

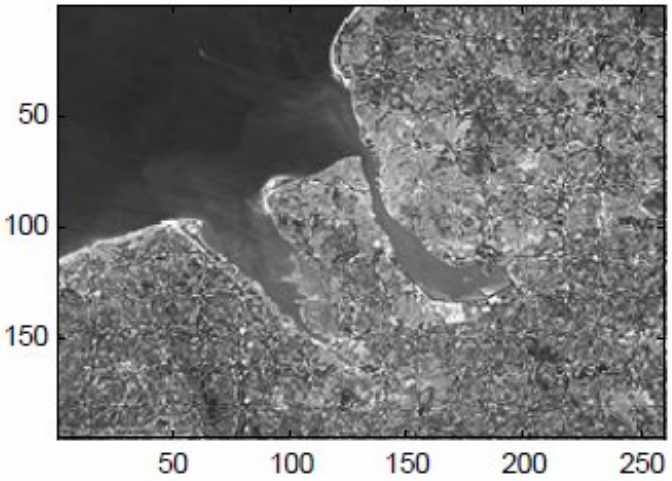
0	1	0
1	-4	1
0	1	0

Liverpool RGB

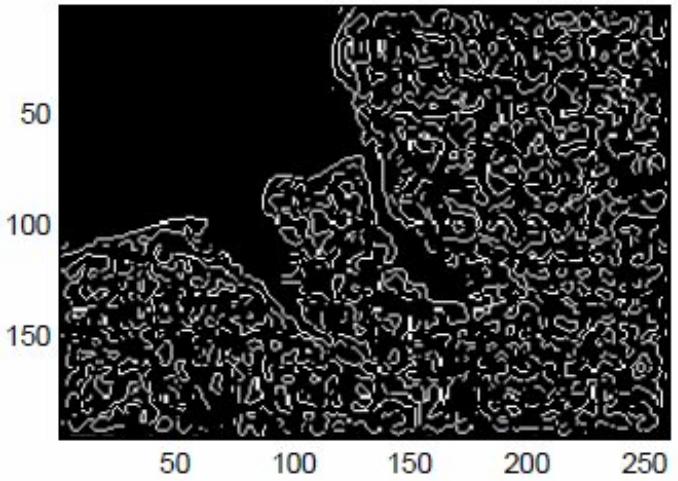
Sobel



Liverpool 256 gris

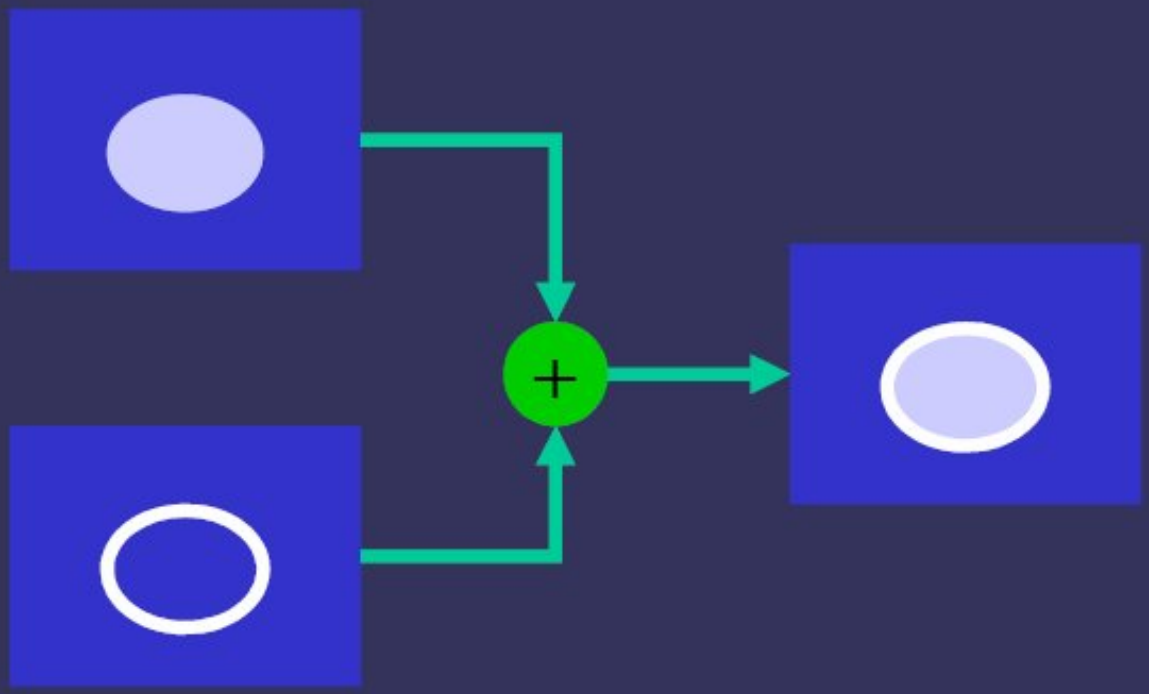


Laplacien



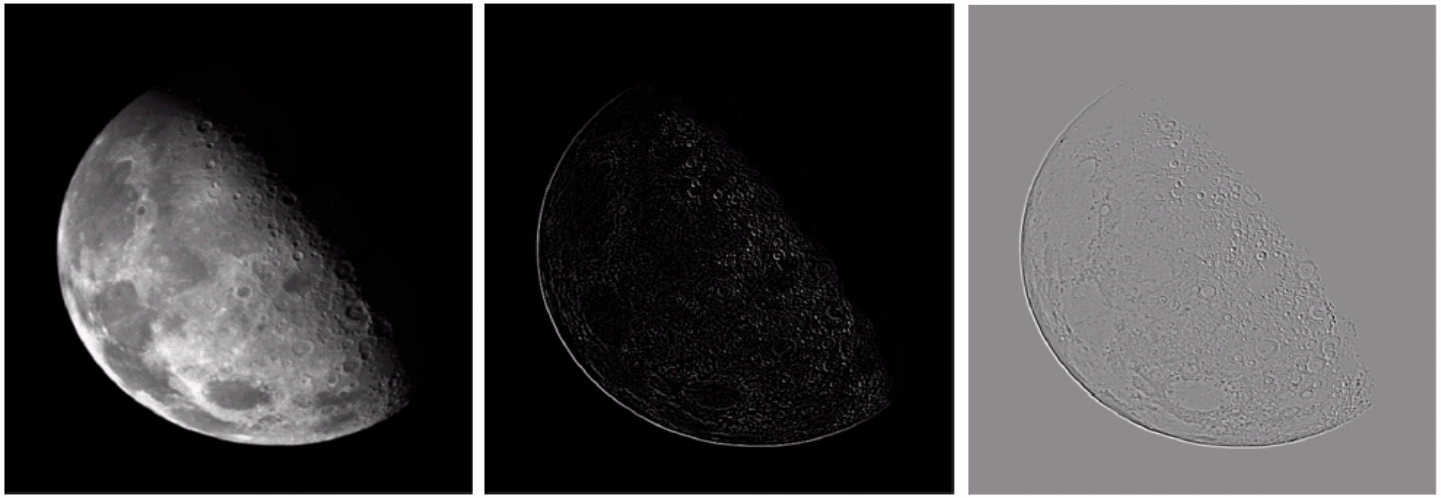
Spatial Filtering

Edges detected by the Laplacian can be used to sharpen the image !



The Laplacian (cont…)

- Applying the Laplacian to an image we get a new image that highlights edges and other discontinuities



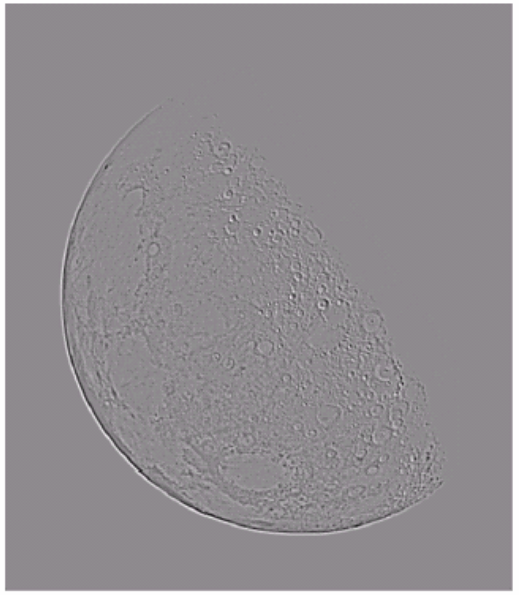
Original Image

Laplacian Filtered Image

Laplacian Filtered Image Scaled for Display

But That Is Not Very Enhanced!

- The result of a Laplacian filtering is not an enhanced image
- We have to do more work in order to get our final image
- Subtract the Laplacian result from the original image to generate our final sharpened enhanced image



Laplacian
Filtered Image
Scaled for Display

$$g(x,y) = f(x,y) - \nabla^2 f$$

Laplacian Image Enhancement

Original Image

Laplacian Filtered Image

Sharpened Image

- In the final sharpened image edges and fine detail are much more obvious

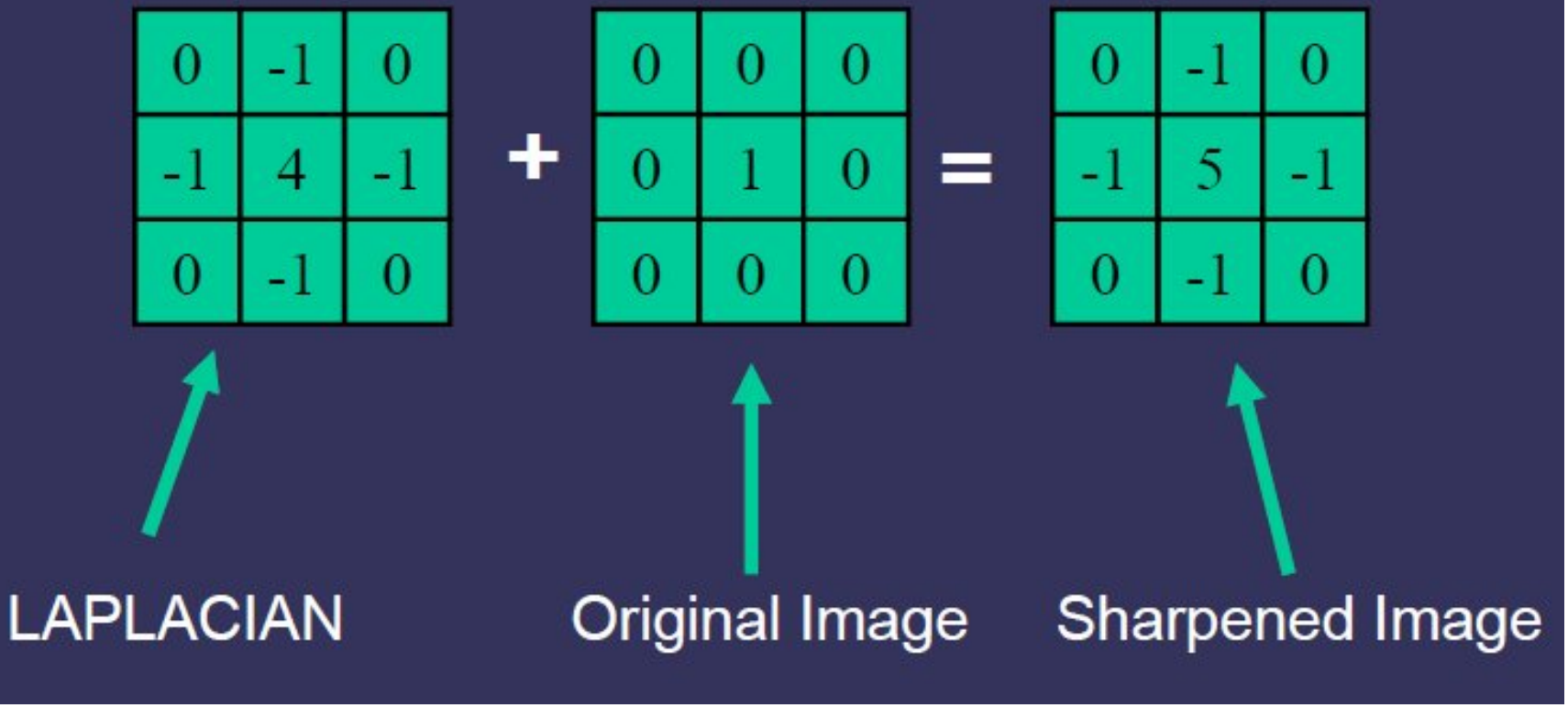
Simplified Image Enhancement

•The entire enhancement can be combined into a single filtering operation

$$
\begin{aligned}
g(x,y) &= f(x,y) - \nabla^2 f \\
&= f(x,y) - [f(x+1,y) f(x-1,y) \\
&\qquad + f(x,y+1) + f(x,y-1)] \\
&\qquad + 4f(x,y) \\
&= 5f(x,y) - [f(x+1,y) f(x-1,y) \\
&\qquad + f(x,y+1) + f(x,y-1)]
\end{aligned}
$$

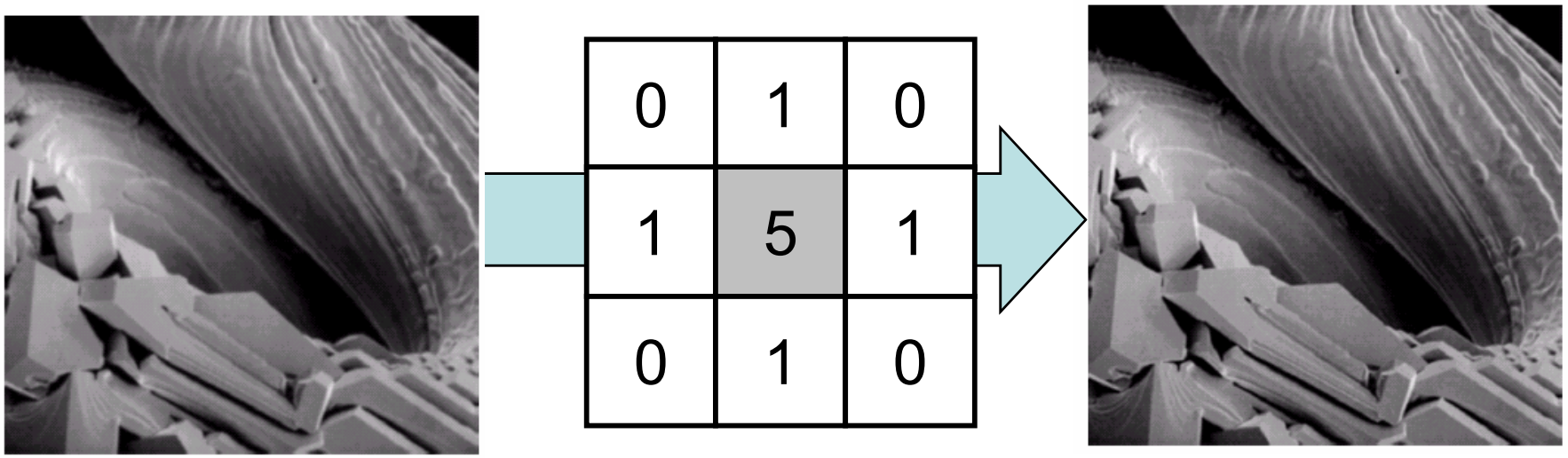
Spatial Filtering

Sharpening can be done in 1 pass:



Simplified Image Enhancement (cont…)

- This gives us a new filter which does the whole job for us in one step



Variants on the Simple Laplacian

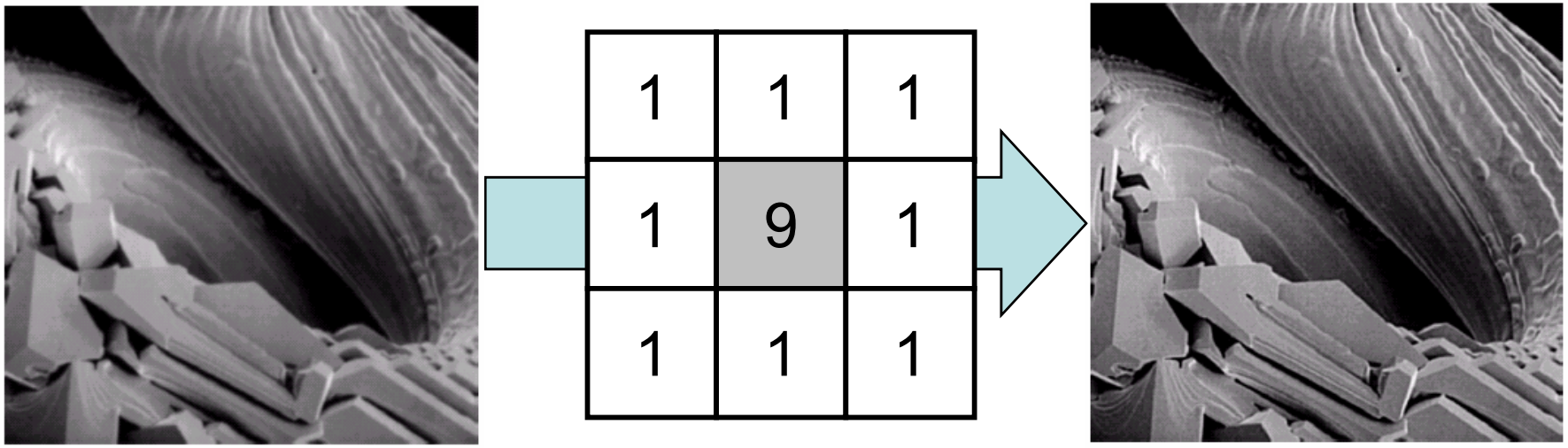
- There are lots of slightly different versions of the Laplacian that can be used:

0	1	0
1	-4	1
0	1	0

Simple Laplacian

1	1	1
1	-8	1
1	1	1

Variant of Laplacian



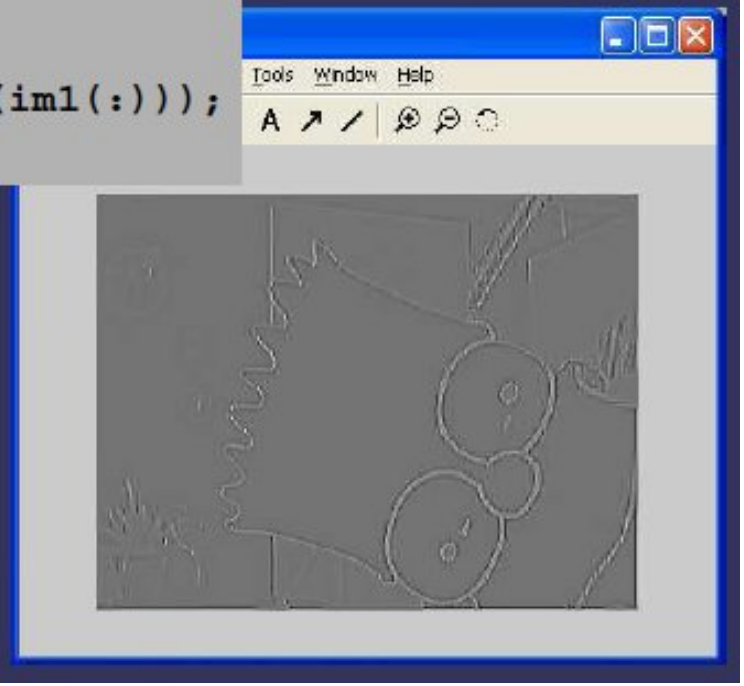
Effect of the Laplacian… (MATLAB demo)

```
im = im/max(im(:));

% create laplacian
L=[1 1 1;1 -8 1; 1 1 1];

% Filter !
im1=conv2(im,L);

% normalize and show
im1=(im1-min(im1(:))) / (max(im1(:))-min(im1(:)));
imshow(im1);
```



Spatial Filtering