Computational Vision & Imaging - Lab 1 Solutions

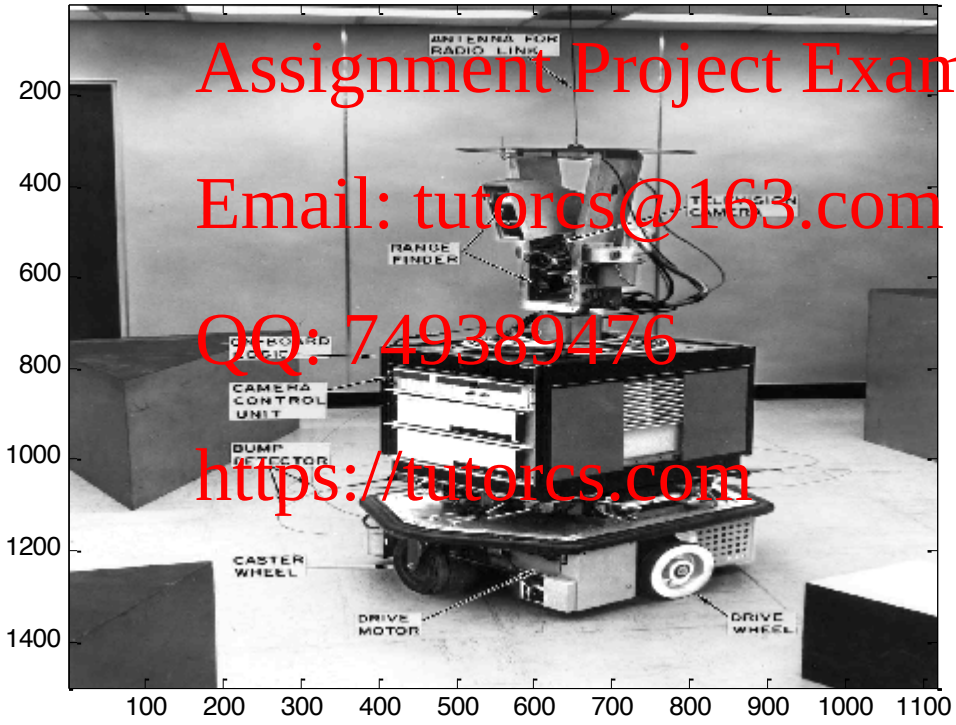
Hamid Dehghani,
School of Computer Science,
University of Birmingham, U.K.
B15 2TT

Common mistakes:

People normally cut a[illegible]nd from the handout and they find that they get errors. One main rea[illegible]nark!

shakey = read_image('', 'shakey.150.gif'); % loads the gray scale image Shakey

show_image(shakey); % Displays image



load sobel % this is a mat file that contains sobelX and sobelY

To do a 2D convolution we use the MATLAB built-in function conv2.

Note: The option 'valid' will output image without zero padding. Type help conv2 for more info

shakey_sobelX = conv2(shakey,sobelX,'valid');

so if you do a whos shakey_sobelX, you will see that the output is 2 pixels smaller in both x and y direction.

They can display new image by:

Show_image(shakey

Question 1:

The task is to write a function magnitude.m should look like this:

function m = magnitu

```
% function m = magnitude(x,y);
% returns the magnitude of two matrices, on an element by element basis

m = sqrt(x.^2 + y.^2);
```


The over effect of changing the threshold is the number of edges found, amount of noise and thickness of the lines: The lower the threshold, the more edges are selected. Optimal value is problem dependent: Note that using this method, you are setting a global threshold: This may not always be a good idea.

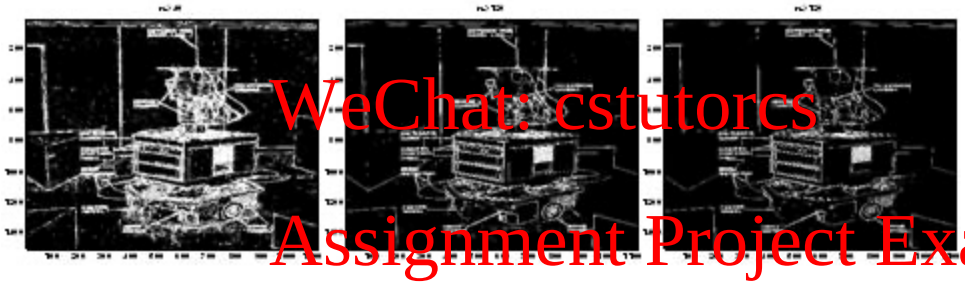
Question 2:

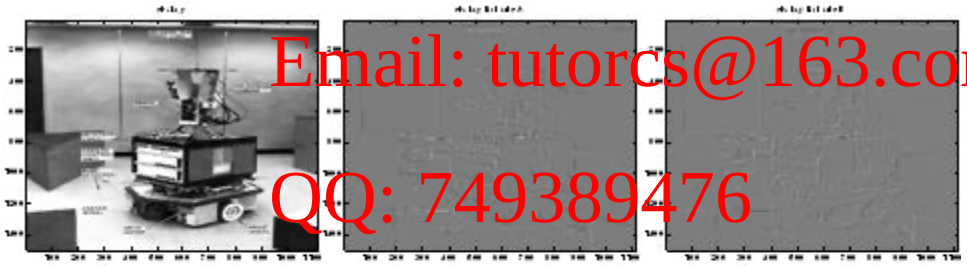
Sobel approximates the gradient in x and y, which are combined to find edges, but Robers is more of a diagonal approximator. This is easily visualized by looking at the images of each gradient. Note that same threshold values will not apply to both edge detectors: They are different operators.

Question 3:

The will also need to write another function, where they create the absolute value and not the magnitude, i.e. m = abs(x) + abs(y);

Difference will be minimal, as with each filter we are already approximating the gradient. But the absolute value is more computationally efficient.





```
function m = absolute(x,y)
% on an element by element basis

m = abs(x) + abs(y);
```