Assignment A3: Enhancement Filters

CS 4640
Fall 2019

Assigned: 6 September 2019

Due: 26 September 2019

For this problem, handin your lab report and the Matlab .m files for the functions described by the headers below.

Some notes:

- Indent headers correctly (5 spaces indented lines)
- Do not exceed 75 characters per source line

None of the functions should write to the interpreter, draw, etc.

Q1. Explore the use of Roberts, Prewitt, Sobel, log, $\mid \nabla f \mid$, zero crossing and Canny on the *franklin* and *metro* images. These will need preprocessing to convert them to gray level images. Discuss the differences of the edge detectors, compare their effectiveness for these images; focus on how they might help delineate words, important shapes, linear features, etc. Make sure to also include the closed contour option in the Matlab *edge* function for the zerocross method. Define performance measures and evaluate the methods with respect to those. Make sure to provide figures to illustrate your discussion. (No Matlab functions required for this question.)

Q2. The text says on p. 105 to "look at the gradient of the zero crossing and only keep zero crossings where this is above a certain threshold (i.e., use the third derivative of the original image)."

Develop filters for $\frac{\partial^3 f}{\partial x^3}$ and $\frac{\partial^3 f}{\partial y^3}$ by extending $\frac{\partial^2 f}{\partial x^2}$ and $\frac{\partial^2 f}{\partial y^2}$ to the third derivative. These should be 1x4 and 4x1 filters. The image *steps.jpg* is provided to explore this and reduces the problem to just the x dimension. Compute the first derivatives, [x1,y1], (using the Matlab *gradient* function), second derivatives, [x2,y2], (using *gradient* on the first derivatives), and the third derivatives, [x3,y3], (using *gradient* on the second derivatives). Compare your third derivative function to the *gradient* function result. Compare them by over-plotting steps(21,:), x1(21,:), x2(21,:), and x3(21,:). Modify your third derivative filters to get the same result as the *gradient* function. On a second figure, over-plot steps(21,:) and the 21st row of the results of applying a Laplacian filter to steps, and running the Matlab *edge* function with the 'log' method. Explain why the 'log' method does not line up with the others. (This question has the CS4640_df3 function.)

Q3. Use all the techniques learned so far to segment:

- line-like features (straight, narrow and long)
- solid rectangular objects
- text objects

Explain the ideas tried for these and provide performance measures. You may want to demonstrate your results on synthetic images with just a few components. (This question has the CS4640_shapes function.)

```
function [dx3,dy3] = CS4640_df3(im)
% CS4640_df3 - third derivative of image in x and y
% On input:
%     im (MxN array): input gray level image
% On output:
%     dx3 (MxN double array): third derivative in x: d^3f/dx^3
%     dy3 (MxN double array): third derivative in y: d^3f/dy^3
% Call:
%     [dx3,dy3] = CS4640_df3(cells);
% Author:
%     <Your name>
%     UU
%     Fall 2019
%
```

```
function segs = CS4640_shapes(im)
% CS4640_shapes - extract simple shapes from image
% On input:
%     im (MxN array): input gray level image
% On output:
%     segs (MxN array): labeled image:
%       0: background or unknown
%       1: line object
%       2: circular object
%       3: text object
% Call:
%     segs = CS4640_shapes(im);
% Author:
%     <Your name>
%     UU
%     Fall 2019
%
```