Assignment A4: Frequency Domain Filtering

CS 4640
Fall 2019

Assigned: 26 September 2019

Due: 17 October 2018

For this problem, handin a report (PDF file) as required below as well as the Matlab .m files for the functions described by the headers below, and any help functions you write.

None of the functions should read or write files, write to the interpreter, draw, plot, etc. unless explicitly required by the header.

1. Develop a texture feature analysis tool based on the 2D FFT power spectrum. For every 5x5 region in the image, compute the 2D FFT, compute the power spectrum, and use that as a 25-element feature vector. Produce a texture feature array, M*N by 25, and then use kmeans as in A2 to explore the usefulness of this for semantic region analysis (based on texture) of document images franklin and metro. Normalize the power spectrum by the value of the (0,0) component and see if this helps. Propose a performance measure (in terms of labeling semantic components) and report results. Develop the function *CS4640_FFT_texture* described below. This function should do the one thing the header says!

```
function T = CS4640_FFT_texture(im)
% CS4640_FFT_texture - compute FFT texture parameters
% On input:
%     im (MxN array): input gray level image
% On output:
%     T (M*Nx25 array): texture parameters
%        each texture parameter is a column vector in T
% Call:
```

```
%     T = CS4640_FFT_texture(im);
% Author:
%     <Your name>
%     UU
%     Fall 2019
%
```

2. Develop a Fourier shape descriptor function as follows:

- Determine connected components
- for every connected component:
  - extract its boundary using bwtraceboundary
  - shift the mean of the points to (0,0)
  - find the principle axes
  - rotate the points so the eigenvector associated with the lower eigenvalue is aligned with the x axis (consider the origin of the image is the lower left corner)
  - (Version 1): treat the x,y boundary points as the real and imaginary parts of a complex number, respectively.
  - (Version 2): compute a function f(p) = $2\pi s(p)/L$, where p is the index into the point set (in the real plane), s(p) is the distance along the polyline from the first point to the p^{th} point, and L is the total length of the polyline.
  - convert the set of points to a fixed standard length, say 50
  - compute the Fourier transform of these complex points (or the f function).
  - treat the resulting vector as a feature vector and add to set of feature vectors.
- use kmeans to cluster all the vectors resulting from connected components to determine different semantic objects in the image.

Explore this idea on the franklin and metro images and report the results in terms of a resonable performance measure. Develop two high-level functions: (1) *CS4640_FFT_C_pts*, and (2) *CS4640_FFT_R_pts*, described below which compute feature vectors based on the complex point formulation and the f function, respectively. These functions should do the one thing their headers say!

```
function X = CS4640_FFT_C_pts(im)
% CS4640_FFT_C_pts - compute Fourier shape descriptors for connected
%                    components in an image treating boundary as
%                    points in the complex plane
% On input:
%     im (MxN array): binary image with objects as foreground
% On output:
%     X (nx50 array): 50 Fourier coefficients per connected component
% Call:
%     X = CS4640_FFT_C_pts(im);
% Author:
%     <Your Name>
%     UU
%     Fall 2019
%
```

```
function X = CS4640_FFT_R_pts(im)
% CS4640_FFT_R_pts - compute Fourier shape descriptors for connected
%                    components in an image based on length curve
% On input:
%     im (MxN array): binary image with objects as foreground
% On output:
%     X (nx50 array): 50 Fourier coefficients per connected component
% Call:
%     X = CS4640_FFT_R_pts(im);
% Author:
%     <Your Name>
%     UU
%     Fall 2019
%
```