**Image and Video Processing**

**Programming Assignment – Report**

**Week3**

% Problem 1

% The script converts a given image into grayscale, detects gradient using

% a sobel filter and thresholds pixel values based on a user specified

% percentile.  
% Convolution and Histogram functions from HW1 are used

img = double(imread('spokenFen.jpg'));

dim = size(img);

Hx = 0.25\*[-1 -2 -1;0 0 0;1 2 1];

Hy = 0.25\*[-1 0 1;-2 0 2;-1 0 1];

fx = computeConv(img,Hx);

fy = computeConv(img,Hy);

mag = zeros(dim(1),dim(2));

for x=1:dim(1)

for y=1:dim(2)

mag(x,y) = sqrt((fx(x,y)^2)+(fy(x,y)^2));

end

end

% thresholding

sortGrad = sort(sort(mag,1,'descend'),2,'descend');

sizeGrad = size(sortGrad);

total = sizeGrad(1)\*sizeGrad(2);

T = input('Enter edge threshold value: ');

threshVal = 0;

breakFlag = 0;

iter = 1;

gradHist = computeHist(mag);

k = dim(1)\*dim(2);

gradHist = sort(gradHist,'descend');

transpose(gradHist);

cumHist = cumsum(gradHist);

cumHist = round(ceil(cumHist/cumHist(length(cumHist))\*100),-1);

for x = 1:length(gradHist)

if cumHist(x) == round(T,-1)

disp('x value');

disp(x)

threshVal = x;

% breakFlag = 1;

break

end

end

% DEBUG

disp(threshVal);

out = zeros(dim(1),dim(2));

for x=1:dim(1)

for y=1:dim(2)

if mag(x,y) >= threshVal(1);

out(x,y) = 255;

else

out(x,y) = 0;

end

end

end

imshow(uint8(out));

imwrite(uint8(out),'spokenFenBW1.jpg');

****

Figure : Spoken Fen by Stanley Donwood



Figure : 90% Thresholding



Figure : 80% Thresholding

% Problem 2

% Script to add noise and filter using median filtering.

noiseLevel=0.01;

inImg = imread('nyc.jpg');

inImg = rgb2gray(inImg);

noisy\_img = imnoise(inImg, 'gaussian', 0, noiseLevel);

subplot(3,1,1),imshow(noisy\_img);subplot(3,1,2),imshow(uint8(medianFiltSquare(double(noisy\_img),3)));subplot(3,1,3),imshow(uint8(medianFiltSquare(double(noisy\_img),5)));

imwrite(noisy\_img,'prog2/noisy.jpg')

imwrite(uint8(medianFiltSquare(double(noisy\_img),3)),'prog2/filter3.jpg')

imwrite(uint8(medianFiltSquare(double(noisy\_img),5)),'prog2/filter5.jpg')

%medianFiltSquare.m

function [ outImg ] = medianFiltSquare( img, dim )

%Convolves the image with the median filter matrix of a given

%dimension

%

filt = ones(dim,dim);

outImg = img;

dimImg = size(img);

dimFilt = size(filt);

imgBoundary = (dimFilt - 1)\*0.5;

disp('Beginning Convolution')

for i = imgBoundary(1) + 1:dimImg(1) - imgBoundary(1)

for j = imgBoundary(2) + 1:dimImg(2) - imgBoundary(2)

temp = 0;

outImg(i,j) = median(median(img(i-imgBoundary(1):i+imgBoundary(1),j-imgBoundary(2):j+imgBoundary(2)).\*filt,1),2);

% wait = waitbar((i\*j)/(dimImg(1)\*dimImg(2)))

end

end

clear wait;

disp('Convolution complete.')

%fin.

end



Figure : Noisy Satellite Picture of NYC



Figure : After 3x3 median filtering



Figure : After 5x5 median filtering

While the 5x5 structural element removes noise to a greater extent (as seen in the river/bay region of the map), it also blurs the entire picture, resulting in loss of detail. Note the smudge of central park boundaries in the 5x5 filtered image as compared to the 3x3 filtered image.