160050030_160050031_160050033_assignment2_Filtering

by Vikrant Garg

Submission date: 21-Aug-2018 02:07AM (UTC+0800)

Submission ID: 991572806

File name: report.txt (14.31K)

Word count: 1244

Character count: 10992

```
function matrix = myLinearContrastStretching(imgmatrix)

[ a, b, dimension] = size(mat2gray(imgmatrix));

if dimension==1

    max1 = max(max(imgmatrix));

    min1 = min(min(imgmatrix));

    matrix = (255/(max1- min1))*(imgmatrix-min1);

else

           max1 = max(max(imgmatrix(:,:,1)));

        min1 = min(min(imgmatrix(:,:,1)));

        matrix(:,:,1) = (255/(max1- min1))*(imgmatrix(:,:,1)-min1);

           max2 = max(max(imgmatrix(:,:,2)));

        min2 = min(min(imgmatrix(:,:,2)));

        matrix(:,:,2) = (255/(max2- min2))*(imgmatrix(:,:,2)-min2);
```

```
      max3 = max(max(imgmatrix(:,:,3)));
    min3 = min(min(imgmatrix(:,:,3)));
    matrix(:,:,3) = (255/(max3- min3))*(imgmatrix(:,:,3)-min3);
end
%% MyMainScript
tic;
%% Your code here
imgPath1 = '../data/lionCrop.mat';
Struct = load(imgPath1);
Image1 = Struct.imageOrig;
InputImage = mat2gray(myLinearContrastStretching(Image1));
radius1=2;
scale1=2;
```

```
OutputImage=myUnsharpMasking(Image1,radius1,scale1);

   figure

subplot(1,2,1);

 imshow(InputImage), colorbar;

 subplot(1,2,2);

 imshow(OutputImage), colorbar;

imgPath2 = '../data/superMoonCrop.mat';

Struct2 = load(imgPath2);

Image2 = Struct2.imageOrig;

InputImage2 = mat2gray(myLinearContrastStretching(Image2));

radius2=2;

scale2=2;

OutputImage2=myUnsharpMasking(Image2,radius2,scale2);
```

```
figure

subplot(1,2,1);

 imshow(InputImage2), colorbar;

 subplot(1,2,2);

 imshow(OutputImage2), colorbar;

toc;

function [OutputImage]= myUnsharpMasking(Image,radius,scale)

A = fspecial('gaussian',[5,5], radius);

convolution = imfilter(Image,A,'conv');

matrix = Image + scale*(Image - convolution);

OutputImage = mat2gray(myLinearContrastStretching(matrix));

end
```

ASSIGNMENT 2 QUESTION 1

Files included --

myLinearContrastStretching.m

myMainScript.m

myUnsharpMasking.m

OUTPUT -

For lionCrop.mat -

Window size of gaussian = 5x5 radius =2 scaling =2

For superMoon.mat

Window size of gaussian = 5x5 radius =2 scaling =2

Ques2

------myMainScript.m--------------------------------

```
%% MyMainScript
tic;
%% Your code here
%Loading barbara original and corrupt it
load('../data/barbara.mat')
im_a = imageOrig;
max1 = max(max(im_a));
min1 = min(min(im_a));
range1 = max1 - min1;
curr_a = normrnd(im_a,0.05*range1);
%loading original grass and noisy
im_b = imread('../data/grass.png');
im_b = double(im_b)/256;
```

```
load('../data/grassNoisy.mat')
curr_b = imgCorrupt;
%loading original honeycomb and noisy
im_c = imread('../data/honeyCombReal.png');
im_c = double(im_c)/256;
load('../data/honeyCombReal_Noisy.mat')
curr_c = imgCorrupt;
figure(1)
[a1,sp_1] = myBilateralFiltering(im_a,curr_a,1.5,9.5);
%%
% [a1_1,~] = myBilateralFiltering(im_a,curr_a,1.5*0.9,9.5);
% [a1_2,~] = myBilateralFiltering(im_a,curr_a,1.5*1.1,9.5);
% [a1_3,~] = myBilateralFiltering(im_a,curr_a,1.5,9.5*0.9);
% [a1_4,~] = myBilateralFiltering(im_a,curr_a,1.5,9.5*1.1);

%%

figure(2)

[a2,sp_2] = myBilateralFiltering(im_b,curr_b,0.9,0.14);

%%

% [a2_1,~] = myBilateralFiltering(im_b,curr_b,0.9*0.9,0.14);

% [a2_2,~] = myBilateralFiltering(im_b,curr_b,0.9*1.1,0.14);

% [a2_3,~] = myBilateralFiltering(im_b,curr_b,0.9,0.14*0.9);

% [a2_4,~] = myBilateralFiltering(im_b,curr_b,0.9,0.14*1.1);

%%

figure(3)

[a3,sp_3] = myBilateralFiltering(im_c,curr_c,1,0.13);

%%
```

```
% [a3_1,~] = myBilateralFiltering(im_c,curr_c,1*0.9,0.13);

% [a3_2,~] = myBilateralFiltering(im_c,curr_c,1*1.1,0.13);

% [a3_3,~] = myBilateralFiltering(im_c,curr_c,1,0.13*0.9);

% [a3_4,~] = myBilateralFiltering(im_c,curr_c,1,0.13*1.1);

%%

figure(4)

imshow(sp_1)

colorbar

figure(5)

imshow(sp_2)

colorbar

figure(6)

imshow(sp_3)
```

```
colorbar

toc;

------------------myBilateralFilter.m------------------

function [RMSD,spg_mat] = myBilateralFiltering(im1,corrupt_im1,ss,si)

%bilateral_filter Summary of this function goes here

% input - Image, sigma-spatial, sigma-intensity

% create corrupt image, apply bilateral filter with given sigmas, calculate

% root squared mean difference between new image and old.

% filename = '../data/barbara.mat';

% ss = 1;

% si = 1;

%load image

%size
```

```
[row1,col1] = size(im1);

%initialize new image

new_im1 = zeros(row1,col1);

for i = 1:row1

    for j = 1:col1

        l = floor(j - 3*ss);

        if (l < 1);  l = 1; end

        r = floor(j + 3*ss);

        if (r > col1);  r = col1; end

        t = floor(i - 3*ss);

        if (t < 1);  t = 1; end

        b = floor(i + 3*ss);

        if (b > row1);  b = row1; end
```

```
X = corrupt_im1(t:b,l:r);
in = X-corrupt_im1(i,j);
sp_r = 1:b-t+1 ;
sp_r = sp_r';
sp_r = repmat(sp_r,1,r-l+1);
sp_c = 1:r-l+1;
sp_c = repmat(sp_c,b-t+1,1);
sp_r = sp_r - (i - t + 1);
sp_c = sp_c - (j - l + 1);
sp_r = sp_r.*sp_r;
sp_c = sp_c.*sp_c;
sp = sp_r + sp_c;
sp = sqrt(sp);
```

```
        in = exp((-0.5/si^2)*(in.*in));
        sp = exp((-0.5/ss^2)*(sp.*sp));
        wts = in.*sp;
        new_im1(i,j) = sum(sum(wts.*X))/sum(sum(wts));
    end
end
%printing gaussian spatial mask
l = 2*floor(3*ss) + 1;
sp_r = 1:l;
sp_r = sp_r';
sp_r = repmat(sp_r,1,l);
sp_c = 1:l;
sp_c = repmat(sp_c,l,1);
```

```
sp_r = sp_r - (floor(3*ss) + 1);

sp_c = sp_c - (floor(3*ss) + 1);

sp_r = sp_r.*sp_r;

sp_c = sp_c.*sp_c;

sp = sp_r + sp_c;

sp = sqrt(sp);

spg_mat = exp((-0.5/ss^2)*(sp.*sp))/(ss*sqrt(2*pi));

RMSD = sqrt(mean(mean((new_im1 - im1).^2)));

% RMSD = 0;

myNumOfColors=200;

myColorScale = [(0:1/(myNumOfColors-1):1)',(0:1/(myNumOfColors-1):1)',(0:1/(myNumOfColors-1):1)'];

subplot(1,3,1)
```

```
imshow(mat2gray(im1))

colormap(myColorScale);

colormap gray;

colorbar

subplot(1,3,2)

imshow(mat2gray(corrupt_im1))

colormap(myColorScale);

colormap gray;

colorbar

subplot(1,3,3)

imshow(mat2gray(new_im1))

colormap(myColorScale);

colormap gray;
```

```
colorbar

end
```

----------Report-2------------------------

Assignment2

Report For Question2

Files Include

1. myMainScript.m

2. myBilateralFiltering.m

How To Run

Run myMainScript.m file from matlab.

OUTPUT

Barbara

Sigma space = 1.5 Sigma intensity = 9.5 RMSD = 3.2827

1. 3.289913177490234

2. 3.282960176467896

3. 3.320384263992310

4. 3.283159971237183Grass

Sigma space = 1 Sigma intensity = 0.14 RMSD = 0.078643311009290

1. 0.078693902682144

2. 0.078715169090018

3. 0.078779692223188

4. 0.078607165451484

HoneyComb

Sigma space = 1 Sigma intensity = 0.13 RMSD = 0.070446266752326

1. 0.070539464802749

2. 0.070475655542278

```
3. 0.070563453428366

4. 0.070447187128614
```

Ques3

```
---------myMainScript.m------------------------------

%% MyMainScript

tic;

%% Your code here

%Loading barbara original and corrupt it

load('../data/barbara.mat')

im_a = imageOrig;

im_a = im_a(1:2:end,1:2:end);
```

```
max1 = max(max(im_a));

min1 = min(min(im_a));

range1 = max1 - min1;

curr_a = normrnd(im_a,0.05*range1);

curr_a = curr_a(1:2:end,1:2:end);

%loading original grass and noisy

im_b = imread('../data/grass.png');

im_b = double(im_b)/256;

load('../data/grassNoisy.mat')

curr_b = imgCorrupt;

%loading original honeycomb and noisy

im_c = imread('../data/honeyCombReal.png');

im_c = double(im_c)/256;
```

```
load('../data/honeyCombReal_Noisy.mat')

curr_c = imgCorrupt;

%%

figure(1)

a1 = myPatchBasedFiltering(im_a,curr_a,2,2);

%%

figure(2)

a2 = myPatchBasedFiltering(im_b,curr_b,1,2);

%%

figure(3)

a3 = myPatchBasedFiltering(im_c,curr_c,0.15,2);

%%

% figure('name','0.001')
% myPatchBasedFiltering('../data/barbara.mat',0.001,1)
% %%
% figure('name','0.01')
% myPatchBasedFiltering('../data/barbara.mat',0.01,1)
% %%
% figure('name','0.1')
% myPatchBasedFiltering('../data/barbara.mat',0.1,1)
% %%
% figure('name','1')
% myPatchBasedFiltering('../data/barbara.mat',1,1)
% %%
% figure('name','10')
% myPatchBasedFiltering('../data/barbara.mat',10,2,1)
% %%

% figure('name','100')

% myPatchBasedFiltering('../data/barbara.mat',100,1)

%%

toc;

-------------myPatchBasedFiltering.m--------------------

function [RMSD,spg_mat] = myPatchBasedFiltering(im1,corrupt_im1,s,ss)

%UNTITLED Summary of this function goes here

%   Detailed explanation goes here

% outputArg1 = inputArg1;

% outputArg2 = inputArg2;

% filename='../data/barbara.mat';

% i=1;
% s=2;

% ss=4;

% load(filename)

% % im1 = imageOrig;

% if i == 1

%    im1 = imageOrig;

% elseif i == 2

%     im1 = imgCorrupt;

% end

% %calculate range

% % im1=im1(1:2:end, 1:2:end);

% max1 = max(max(im1));

% min1 = min(min(im1));
% range1 = max1 - min1;

%size

[row1,col1] = size(im1);

%corrupting image with gaussian noise with std dev = 5% of range

% corrupt_im1 = normrnd(im1,0.05*range1);

%initialize new image

new_im1 = zeros(row1,col1);

windows=25;

patchs=9;

w=(windows-1)/2;

p=(patchs-1)/2;

for i=1:row1

    if((i-w)>=1)
    iwmin=i-w;
else
    iwmin=1;
end
if((i+w)<=row1)
    iwmax=i+w;
else
    iwmax=row1;
end
if((i-p)>=1)
    ipmin=i-p;
else
    ipmin=1;
```

```
end

if((i+p)<=row1)

    ipmax=i+p;

else

    ipmax=row1;

end

for j=1:col1

    if((j-w)>=1)

        jwmin=j-w;

    else

        jwmin=1;

    end

    if((j+w)<=col1)
    jwmax=j+w;
else
    jwmax=col1;
end
if((j-p)>=1)
    jpmin=j-p;
else
    jpmin=1;
end
if((j+p)<=col1)
    jpmax=j+p;
else
    jpmax=col1;
```

```
end

% patch((ipmin))=im1(ipmin:ipmax, jpmin:jpmax);

% for ip1=ipmin:ipmax

%     for jp1=jpmin:jpmax

%         patch((ip1-ipmin+1),(jp1-jpmin+1))=im1(ip1,jp1);

%     end

% end

wt = zeros(windows,windows);

spacial=zeros(windows,windows);

window_image=zeros(windows,windows);

         sp_r = 1:row1;

         sp_r = sp_r';

         sp_r = repmat(sp_r,1,col1);
```

```
          sp_c = 1:col1;

          sp_c = repmat(sp_c,row1,1);

          sp_r = sp_r - i;

          sp_c = sp_c - j;

          sp_r = sp_r.*sp_r;

          sp_c = sp_c.*sp_c;

          sp = sp_r + sp_c;

          % sp = sqrt(sp);

    % lambda=jwmax-jwmin+1;

%     sum_wt=0;

%     numerator=0;

    for iw=iwmin:iwmax

      if((iw-p)>=1)
    ipmin1=iw-p;
else
    ipmin1=1;
end
if((iw+p)<=row1)
    ipmax1=iw+p;
else
    ipmax1=row1;
end
for jw=jwmin:jwmax
    if((jw-p)>=1)
        jpmin1=jw-p;
    else
    jpmin1=1;

end

if((jw+p)<=col1)

    jpmax1=jw+p;

else

    jpmax1=col1;

end

% (1:ipmax-ipmin+1, 1:jpmax-ipmin+1)

% (1:ipmax1-ipmin1+1,1:jpmax1-jpmin1+1)

if ipmax-ipmin==9 && jpmax-jpmin==9

    %display(i);

    %display(iw);

    %display(ipmin1);
```

```
            %display(ipmax1);
            %display
            patch1=corrupt_im1(i-(iw-ipmin1):i+(ipmax1-iw),  j-(jw-jpmin1):j+(jpmax1-jw));
            patch2=corrupt_im1(ipmin1:ipmax1,jpmin1:jpmax1);
        else
            a=ipmax-ipmin;
            a1=ipmax1-ipmin1;
            b=jpmax-jpmin;
            b1=jpmax1-jpmin1;
            if a>a1
                if b>b1
                    patch1=corrupt_im1(i-(iw-ipmin1):i+(ipmax1-iw), j-(jw-jpmin1):j+(jpmax1-jw));
```

```
    patch2=corrupt_im1(ipmin1:ipmax1, jpmin1:jpmax1);

    c=sp(i-(iw-ipmin1):i+(ipmax1-iw), j-(jw-jpmin1):j+(jpmax1-jw));

  else

    patch1=corrupt_im1(i-(iw-ipmin1):i+(ipmax1-iw), jpmin:jpmax);

    patch2=corrupt_im1(ipmin1:ipmax1, jw-(j-jpmin):jw+(jpmax-j));

    c=sp(i-(iw-ipmin1):i+(ipmax1-iw), jpmin:jpmax);

  end

else

  if b>b1

    patch1=corrupt_im1(ipmin:ipmax,    j-(jw-jpmin1):j+(jpmax1-jw));

    patch2=corrupt_im1(iw-(i-ipmin):iw+(ipmax-i),jpmin1:jpmax1);

    c=sp(ipmin:ipmax,j-(jw-jpmin1):j+(jpmax1-jw));

  else
```

```
              patch1=corrupt_im1(ipmin:ipmax, jpmin:jpmax);

              patch2=corrupt_im1(iw-(i-ipmin):iw+(ipmax-i), jw-(j-jpmin):jw+(jpmax-
j));

              c=sp(ipmin:ipmax, jpmin:jpmax);

            end

          end

        end

        % for ip2=ipmin1:ipmax1

        %     for jp2=jpmin1:jpmax1

        %         patch2((ip2-ipmin1+1),(jp2-jpmin2+1))=im1(ip2,jp2);

        %     end

        % end

        patch3=patch1-patch2;
```

```
                c=(1/(sqrt(2*pi)*ss))*exp((-1/(2*ss*ss))*c);

                wt(iw-iwmin+1,jw-jwmin+1)=exp(-(sumsqr(patch3.*c)/s*s));

                  % spacial(iw-iwmin+1, jw-jwmin+1)=(i-iw)*(i-iw)+(j-jw)*(j-jw);

                  window_image(iw-iwmin+1, jw-jwmin+1)=corrupt_im1(iw, jw);

%                 sum_wt=sum_wt+wt;

%                 numerator()=numerator+wt*corrupt_im1(iw,jw);

              end

          end

          % sp = exp((spacial*(-0.5))/(ss*ss));

%         n_wt=exp((-1/s*s)*wt);

%         fprintf()

%         save('special','sp');

%         save('weights','wt');
```

```
        % n_wt = sp.*wt;
%           n_wt=wt;
        new_im1(i,j)=sum(sum(window_image.*wt))/sum(sum(wt));
    end
%       fprintf('%d of %d ₩n', i,row1);
end
%printing gaussian spatial mask
l = 2*floor(3*ss) + 1;
sp_r = 1:l;
sp_r = sp_r';
sp_r = repmat(sp_r,1,l);
sp_c = 1:l;
sp_c = repmat(sp_c,l,1);
```

```
sp_r = sp_r - (floor(3*ss) + 1);

sp_c = sp_c - (floor(3*ss) + 1);

sp_r = sp_r.*sp_r;

sp_c = sp_c.*sp_c;

sp = sp_r + sp_c;

sp = sqrt(sp);

spg_mat = exp((-0.5/ss^2)*(sp.*sp))/(ss*sqrt(2*pi));

RMSD = sqrt(mean(mean((new_im1 - im1).^2)));

display(RMSD)

myNumOfColors=200;

myColorScale = [(0:1/(myNumOfColors-1):1)',(0:1/(myNumOfColors-1):1)',(0:1/(myNumOfColors-1):1)'];

subplot(1,3,1)
```

```
imshow(mat2gray(im1))

colormap(myColorScale);

colormap gray;

colorbar

subplot(1,3,2)

imshow(mat2gray(corrupt_im1))

colormap(myColorScale);

colormap gray;

colorbar

subplot(1,3,3)

imshow(mat2gray(new_im1))

colormap(myColorScale);

colormap gray;
```

colorbar

end

-----------Report 3-------------------

Assignment 2 Question 3

Files Included

1. myMainScript.m

2. myPatchBasedFiltering.m

160050030_160050031_160050033_assignment2_Filtering

ORIGINALITY REPORT

12%	6%	9%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	amaliepews.blogg.no Internet Source	5%
2	Submitted to Indian Institute of Technology, Bombay Student Paper	2%
3	James F. Peters. "Topology of Digital Images", Springer Nature America, Inc, 2014 Publication	2%
4	Nurullah Ankaralıoğlu, Hüseyin Aydın, Inci Gültekin. "Computation of abstract groups S((d+1)/2,d) and torus knots K(d,2)", Applied Mathematics and Computation, 2005 Publication	1%
5	Complex Contact and Symmetric Manifolds, 2005. Publication	1%
6	Submitted to National University of Ireland, Galway Student Paper	1%

Manaf Adnan Saleh Saleh. "New types of

7	Lipschitz summing maps between metric spaces", Mathematische Nachrichten, 2017 Publication	<1%
8	A. KOHZU. "Natural 13C and 15N abundance of field-collected fungi and their ecological implications", New Phytologist, 11/1999 Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

160050030_160050031_160050033_assignment2_Filtering

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21

PAGE 22

PAGE 23

PAGE 24

PAGE 25

PAGE 26

PAGE 27

PAGE 28

PAGE 29

PAGE 30

PAGE 31

PAGE 32

PAGE 33

PAGE 34

PAGE 35

PAGE 36

PAGE 37

PAGE 38

PAGE 39