Marking Scheme

Exam Paper
BSc CS

Digital Image Processing (0750474)

Final exam  Second semester  Date: 28/05/2012

Section 1
Weighting 40% of the module total

Lecturer:  Dr. Qadri Hamarsheh
Coordinator:  Dr. Qadri Hamarsheh
Internal Examiner:  Dr. Nameer N. EL-Emam
Marking Scheme

Digital Image Processing (0750474)

The presented exam questions are organized to overcome course material through 6 questions. The all questions are compulsory requested to be answered.

Marking Assignments

Question 1: This question is attributed with 7 marks if answered properly; the answers are as following:

1) Computer vision defined as a discipline in which --------------
   a) Both the input and output of a process are images.
   b) The input of a process is an image description and the output is image
   c) Both the input and output of a process are descriptions.
   d) The input of a process is an image and the output is an image description.

2) MRI Technology used --------- band.
   a) Gamma Rays
   b) CT Scan
   c) Visible
   d) Radio waves

3) Filter cannot be implemented using convolution mechanism.
   a) Average
   b) Gaussian
   c) Median
   d) Disk

4) To remove "salt-and-pepper" noise without blurring we use
   a) Max Filter
   b) Median Filter
   c) Min Filter
   d) Smoothing Filter

5) Edge detection in images is commonly accomplished by performing a spatial ---
   --- of the image field.
   a) Smoothing Filter
   b) Integration
   c) Differentiation
   d) Min Filter

6) Both the -------- and -------- filters are used to enhance horizontal edges (or vertical if transposed).
   a) Prewitt and Sobel
   b) Sobel and Gaussian
   c) Prewitt and Laplacian
   d) Sobel and Laplacian

7) Transforming the pixel values of an image using the log ( ) transformation is an example of contrast compression of the dark pixels
   a) True
   b) False

Question 2: This question is attributed with 13 marks if answered properly, the answers are as following:

Solution

Question 2.a (2 marks)

• Boundary Representation
• Region Representation

Question 2.b (3 marks)

a) 4-connectivity:
   Two pixels p and q with values from V are 4-connected if q is in the set N4 (p)

b) 8-connectivity:
   Two pixels p and q with values from V are 8-connected if q is in the set N8 (p)

c) m-connectivity:
   Two pixels p and q with values from V are m-connected if
   i. q is in N4 (p) or
   ii. q is in ND (p) and the set N4(p) ∩ N4(q) ≠ Θ

Question 2.c (3 marks)

1) Perwitt filter.
2) Roberts filter
3) Sobel filter.
4) Unsharp filter

Question 2.d (2 marks)

D₃ Distance (city block distance) is defined by

\[ D₃(p, q) = |x - s| + |y - t| \]

D₈ Distance (chess board distance) is defined by
Unsharp masking:
The process of subtracting an unsharp (smoothed) version of an image from the original images to get sharpened images (high pass filtering).

The process consists of the following steps:
   a) Blur the original image.
   b) Subtract the blurred image from the original (the resulting difference is called the mask).
   c) Add the mask to the original image.

Unsharp masking is expressed in equations form as follows:

\[
g(x, y) = f(x, y) - \tilde{f}(x, y)
g(x, y) = f(x, y) + k \cdot \delta_{\text{mask}}(x, y)
\]

\(\tilde{f}(x, y)\) – Blurred image, \(f(x, y)\) – original image, \(\delta_{\text{mask}}(x, y)\) – unsharp mask.

Then we add a weighted portion of the mask back to the original image:

\[
f(x, y) = f(x, y) - \tilde{f}(x, y) + k \cdot \delta_{\text{mask}}(x, y)
\]

**Question 2:**

Unsharp masking:
The process of subtracting an unsharp (smoothed) version of an image from the original images to get sharpened images (high pass filtering).

The process consists of the following steps:
   a) Blur the original image.
   b) Subtract the blurred image from the original (the resulting difference is called the mask).
   c) Add the mask to the original image.

Unsharp masking is expressed in equations form as follows:

\[
g(x, y) = f(x, y) - \tilde{f}(x, y)
g(x, y) = f(x, y) + k \cdot \delta_{\text{mask}}(x, y)
\]

\(\tilde{f}(x, y)\) – Blurred image, \(f(x, y)\) – original image, \(\delta_{\text{mask}}(x, y)\) – unsharp mask.

Then we add a weighted portion of the mask back to the original image:

\[
f(x, y) = f(x, y) - \tilde{f}(x, y) + k \cdot \delta_{\text{mask}}(x, y)
\]

**Question 3:**

This question is attributed with 4 marks if answered properly, the answers are as following:
The complete code for this question as the following:

```matlab
f = imread('pout.tif');
h = imhist(f);
h1 = h(1:10:256);
horiz = 1:10:256;
bar(horiz,h1); (2 marks)
xlabel('Intesity Level','fontsize',12);
ylabel('Count of Pixels','fontsize',12);
title('Image Histogram');
axis([0 255 0 1500]);
set(gca,'xtick',[0:50:255]);
set(gca,'ytick',[0:200:1500]); (2 marks)
```

**Question 4:**

This question is attributed with 4 marks if answered properly, the answers are as following:
The complete code for this question as the following:

```matlab
f = imread('pout.tif');
f = im2double(f); (1 mark)
m= input('Enter the value of m  ');
E= input('Enter the value of m  ');
g = 1./(1+(m./(f+eps)).^E); (2 marks)
figure,imshow(f),title('Original Image');
figure,imshow(g),
title('Contrast stretched Image'); (1 mark)
```

**Question 5:**

This question is attributed with 6 marks if answered properly, the answers are as following:
The complete code for this question as the following:

```matlab
function [mx, mm]=immaxmin(f)
%IMMAXMIN Computes the new Max and Min images
%[MX, MM]=IMMAXMIN(F) Computes MX and MM images
%obtained using 3x3 operations for each pixel
%one operation: the max of the 3x3 pixel environment
%and the second operation is the min of the 3x3).
%MX - the image obtained from the MAX Filter processing.
%MM - the image obtained from the MIN Filter processing.
[M N d] = size(f);
%preallocating arrays for images
mx = uint8(zeros(M,N));
mm = uint8(zeros(M,N)); (2 marks)
%using low-level processing
for x= 2:M-1
```

\(D_{(p, q)} = \max (|x - s|, |y - t|)\).

**Question 2.e**

Unsharp masking:
The process of subtracting an unsharp (smoothed) version of an image from the original images to get sharpened images (high pass filtering).

The process consists of the following steps:
   a) Blur the original image.
   b) Subtract the blurred image from the original (the resulting difference is called the mask).
   c) Add the mask to the original image.

Unsharp masking is expressed in equations form as follows:

\[
g(x, y) = f(x, y) - \tilde{f}(x, y)
g(x, y) = f(x, y) + k \cdot \delta_{\text{mask}}(x, y)
\]

\(\tilde{f}(x, y)\) – Blurred image, \(f(x, y)\) – original image, \(\delta_{\text{mask}}(x, y)\) – unsharp mask.

Then we add a weighted portion of the mask back to the original image:

\[
f(x, y) = f(x, y) - \tilde{f}(x, y) + k \cdot \delta_{\text{mask}}(x, y)
\]
for y=2:N-1
fr = [f(x-1,y-1) f(x-1,y) f(x-1,y+1)];
sr = [f(x,y-1) f(x,y) f(x,y+1)];
tr = [f(x+1,y-1) f(x+1,y) f(x+1,y+1)];
mafm = [fr sr tr];
mx(x,y) = max(mafm);
mm(x,y)= min(mafm);
end;
end;

imshow(mx), figure,
imshow(mm), figure, imshow(f);

Question 6: This question is attributed with 6 marks if answered properly, the answers are as following:
The complete code for this question as the following:

function Noise_EdgeFilter (filename)
%Read and display the input image
f = imread(filename);
f = im2double(f);
subplot(2,2,1);
imshow(f,[],);
title('Input Image');

%Noise Reduction using Gaussian filter
h1 = fspecial('gaussian',[5 5],0.5); %create the Gaussian mask
g1 = imfilter(f,h1,'replicate');
subplot(2,2,2);
imshow(g1,[],);
title('image using Gaussian filter');

%Edge Detection using Laplacian filter
h2 = fspecial('laplacian',0.5);
g2 = imfilter(g1,h2,'replicate');
subplot(2,2,3);
imshow(g2,[],);
title('Laplacian image');

%Noise Reduction using Gaussian filter
h1 = fspecial('gaussian',[5 5],0.5); %create the Gaussian mask
g1 = imfilter(f,h1,'replicate');
subplot(2,2,4);
imshow(g3,[],);
title('image using Laplacian filter');

for y=2:N-1
fr = [f(x-1,y-1) f(x-1,y) f(x-1,y+1)];
sr = [f(x,y-1) f(x,y) f(x,y+1)];
tr = [f(x+1,y-1) f(x+1,y) f(x+1,y+1)];
mafm = [fr sr tr];
mx(x,y) = max(mafm);
mm(x,y)= min(mafm);
end;
end;

imshow(mx), figure,
imshow(mm), figure, imshow(f);