Midterm Solution

Problem 1: *10pt*

Solution:

a) In a color camera, similar to the cones of human visual system, there is a separate sensor sensitive to each of the three primary colors (R, G and B) that records this component.

b) A color monitor displays color images by having three phosphors at each pixel, each generating a different primary color (red, green, blue).

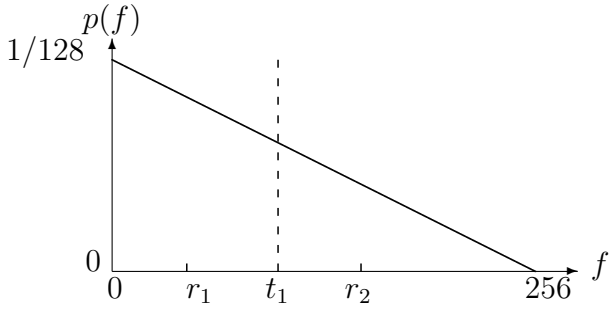
c) Color printer prints color images by having three color inks (cyan, magenta, yellow). A color image stored in RGB format (or other formats) is first converted to the CMY coordinate, and the small area corresponding to each pixel is covered with the amount of cyan, magenta, and yellow inks proprotional to the CMY values of that pixel.

d) Compressing in the YCbCr format make use of two properties. First, the Cb and Cr components in natural images tend to have significantly narrower bandwidth than the Y component, and therefore they can be sampled at a lower rate, typically half of that for Y component in both horizontal and vertical directions. So compared to RGB representations, where all color components have the same number of samples, the YCbCr representation would have 1/4 of the number of samples for the Cb and Cr components respectively. Specifically for an image of size M*N, you would have 3MN values with RGB, whereas with YCbCr representation you could use only MN+1/4(MN)+1/4(MN)=1.5 MN. Second, the human visual system is more sensitive to luminance component than the chrominance components, therefore, we could introduce more distortions to the Cb and Cr components (ie. using less number of bits per sample) without causing visible artifacts. e) A camera usually captures the image in the RGB format. A JPEG image coder first converts it to YCbCr format. It then down samples the Cb and Cr components, and compresses the Y, Cb, and Cr components separately. Before printing this image, a JPEG decoder first decompresses the Y, Cb, Cr components, upsamples the Cb and Cr components, and then converts the YCbCr image to CMY format. The printer then prints the CMY components.

Problem 2: *10pt*

Solution:

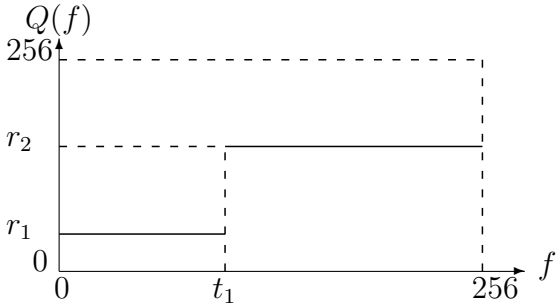
Because the signal is limited to between 0 and 256, we have $t_0 = 0$, $t_2 = 256$ and likely positions of t_1 and r_1 and r_2 are indicated in the figure below.



The quantizer function can be written as

$$Q(f) = \begin{cases} r_1 & \text{if } 0 \le f \le t_1, \\ r_2 & \text{if } t_1 < f \le 256, \end{cases}$$

which is illustrated below.



From the given plot for the pdf, the pdf can be expressed as:

$$p(f) = 1 - \frac{1}{128}\frac{f}{256}\ ,\ 0 \le f \le 256$$

Therefore, the quantization error is

$$\begin{aligned} MSE = \sigma_q^2 &= \int_{-\infty}^{\infty} [f - Q(f)]^2 p(f) \mathrm{d}f \\ &= \int_0^{t_1} (f - r_1)^2 p(f) \mathrm{d}f + \int_{t_1}^{256} (f - r_2)^2 p(f) \mathrm{d}f \end{aligned}$$

To determine r_1, r_2 and t_1, we have to minimize the MSE. This can be accom-

plished by setting the following derivatives to zero:

$$\frac{\partial \sigma_q^2}{\partial t_1} = 0 \quad (1)$$

$$\frac{\partial \sigma_q^2}{\partial r_1} = 0 \quad (2)$$

$$\frac{\partial \sigma_q^2}{\partial r_2} = 0 \quad (3)$$

Eq (1) yields

$$(t_1 - r_1)^2 p(t_1) = (t_1 - r_2)^2 p(t_1) \rightarrow (t_1 - r_1)^2 = (t_1 - r_2)^2 \quad (4)$$

The above has two possible outcomes:

$$(t_1 - r_1) = (t_1 - r_2) \rightarrow \text{ No legitimate solution} \quad (5)$$

$$(t_1 - r_1) = -(t_1 - r_2) \rightarrow t_1 = \frac{r_1 + r_2}{2}.\text{(A legitimate solution!)} \quad (6)$$

Eq (2) yields

$$\int_0^{t_1} (f - r_1) p(f) df = 0 \rightarrow r_1 = \frac{\int_0^{t_1} f p(f) df}{\int_0^{t_1} p(f) df} \quad (7)$$

Similarly, Eq (3) yields

$$\int_{t_1}^{256} (f - r_2) p(f) df = 0 \rightarrow r_2 = \frac{\int_{t_1}^{256} f p(f) df}{\int_{t_1}^{256} p(f) df} \quad (8)$$

Solving the above three equations (6,7,8) will yield

$$t_1 = 128(3 - \sqrt{5}) \approx 97.7833, r_1 \approx 45.0444, r_2 \approx 150.5222.$$

With above t_1, r_1 and r_2, $MSE = 1014.5$.

Note that you only need to express your solutions in terms of Equations (6,7,8) to get full points in this problem.

Problem 3: *15pt*
Solution:

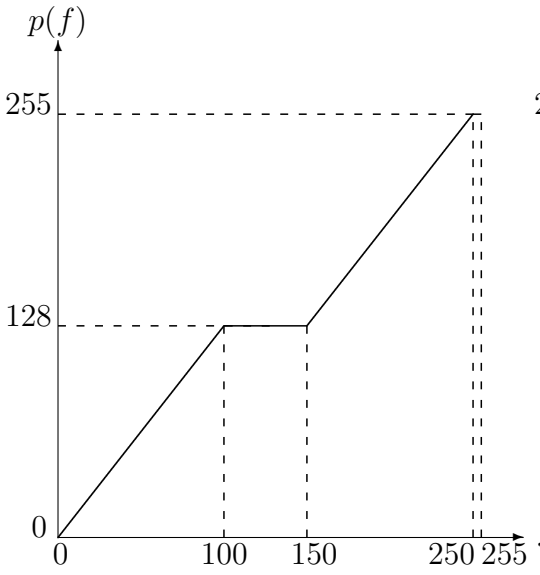


Figure (a)

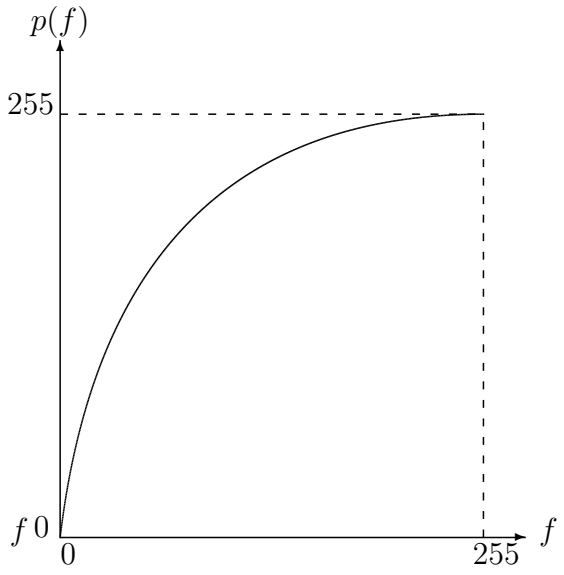


Figure (b)

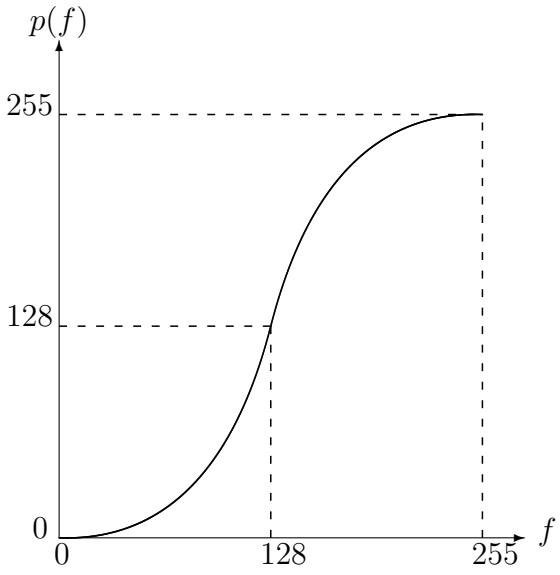
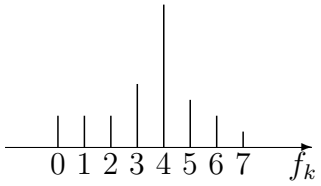


Figure (c)

Problem 4: *10pt*

Solution:

Histogram of original image:



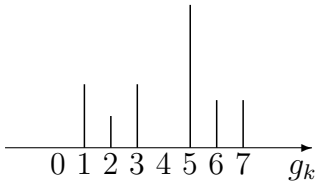
We have:

f_k	$p_F(k)$	$\tilde{g}(k) = \sum_0^k p_F(k)$	$[\tilde{g}(k) \times 7]$	$p_G(k)$	g_k
0	2/25	2/25	$0.56 \approx 1$	0	0
1	2/25	4/25	$1.12 \approx 1$	4/25	1
2	2/25	6/25	$1.68 \approx 2$	2/25	2
3	4/25	10/25	$2.8 \approx 3$	4/25	3
4	9/25	19/25	$5.32 \approx 5$	0	4
5	3/25	22/25	$6.16 \approx 6$	9/25	5
6	2/25	24/25	$6.72 \approx 7$	3/25	6
7	1/25	1	7	3/25	7

The transformation function is:

f	0	1	2	3	4	5	6	7
g	1	1	2	3	5	6	7	7

Histogram of processed image:



The processed image looks like:

1	1	1	3	5
1	2	3	5	5
2	3	5	5	6
3	5	5	6	7
5	5	6	7	7

Problem 5: *10pt*

Solution:

1)

$$H_1 = \frac{1}{8}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 12 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

This filter is not separable. Because there are both positive and negative coefficients and the sum of coefficients is 1, the functionality of this filter is **Image Sharpening**.

2)

$$H_2 = \frac{1}{4}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}\begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$$

This filter is separable as shown above. Because there are both positive and negative coefficients and the sum of coefficients is 0. The functionality of this filter is **Edge Detection**. To be more specific, the vertical filter is a smoothing filter and the horizontal filter is an edge detector. Therefore, this filter is designed to detect vertical edges.

3)

$$H_3 = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} = \frac{1}{16}\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}\begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

This filter is separable as shown above. Because there are only positive coefficients and they sum to 1, the functionality of this filter is **Image Smoothing**.

Problem 6: *5pt*

Solution:

a)

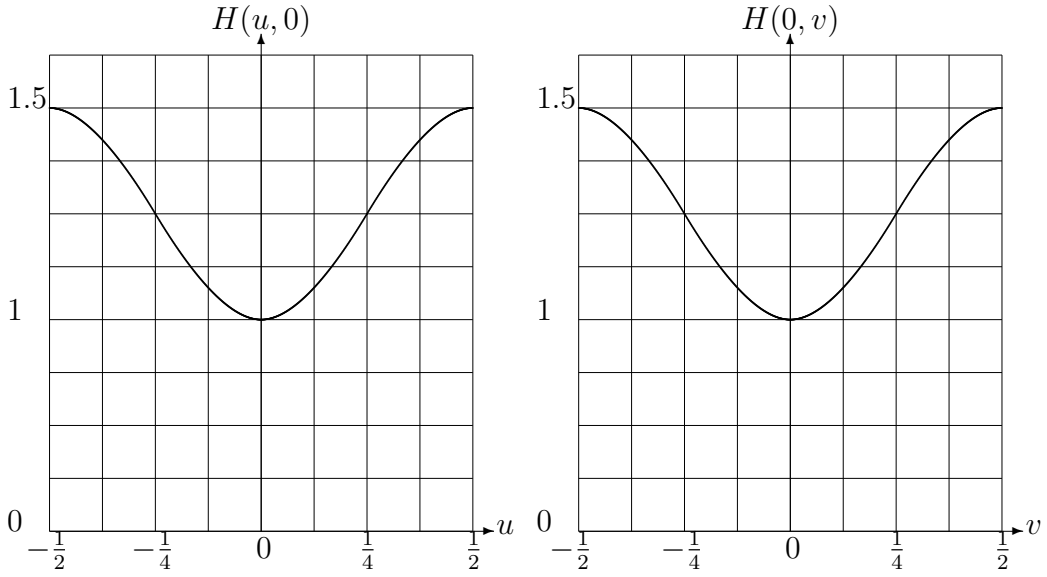
$$F(u,v) = DTFT[f(m,n)] = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} f(m,n)e^{-j2\pi(mu+nv)}$$

$$\begin{aligned} H_1(u,v) &= \sum_{m=-1}^{1}\sum_{n=-1}^{1} H_1(m,n)e^{-j2\pi(mu+nv)} \\ &= \frac{1}{8}\Big[-e^{-j2\pi(-u)} - e^{-j2\pi(-v)} + 12 - e^{-j2\pi(u)} - e^{-j2\pi(v)}\Big] \\ &= \frac{1}{8}\Big[12 - 2\cos(2\pi u) - 2\cos(2\pi v)\Big] \end{aligned}$$

Then,

$$\begin{aligned} H_1(u,0) &= \frac{1}{8}\Big[10 - 2\cos(2\pi u)\Big] \\ H_1(0,v) &= \frac{1}{8}\Big[10 - 2\cos(2\pi v)\Big] \end{aligned}$$

One dimensional profiles:



b) The frequency response indicate this is a **High Emphasis** filter for both u and v dimension.

Problem 7: *15pt*
Solution:

$$F = \begin{bmatrix} 100 & 100 & 0 & 0 \\ 100 & 100 & 0 & 0 \\ 100 & 100 & 0 & 0 \\ 100 & 100 & 100 & 100 \end{bmatrix}, \; H_x = \frac{1}{2}\begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}, \; H_y = \frac{1}{2}\begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix}$$

Note that the **origin** of the filters H_x and H_y are both at the **top-left** corner.

Apply symmetric extension to F:

$$\hat{F} = \left[\begin{array}{c|ccc|cc} 100 & 100 & 100 & 0 & 0 & 0 \\ \hline 100 & 100 & 100 & 0 & 0 & 0 \\ 100 & 100 & 100 & 0 & 0 & 0 \\ 100 & 100 & 100 & 0 & 0 & 0 \\ 100 & 100 & 100 & 100 & 100 & 100 \\ \hline 100 & 100 & 100 & 100 & 100 & 100 \end{array}\right]$$

The central 4×4 segment represents original F.

To perform filtering, H_x and H_y should be flipped properly:

$$\hat{H}_x = \frac{1}{2}\begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix}, \; \hat{H}_y = \frac{1}{2}\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}$$

Note that the **origin** of the flipped filters $\hat{H}_x$ and $\hat{H}_y$ are located at the **bottom-right** corner.

$$G_x = F * \hat{H}_x = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 50 & 100 \end{bmatrix}$$

Similarly,

$$G_y = F * \hat{H}_y = \begin{bmatrix} 0 & 0 & -100 & 0 \\ 0 & 0 & -100 & 0 \\ 0 & 0 & -100 & 0 \\ 0 & 0 & -50 & 0 \end{bmatrix}$$

For $G_m = |G_x| + |G_y|$,

$$G_m = \begin{bmatrix} 0 & 0 & 100 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 100 & 100 \end{bmatrix}$$

Apply $T = 25$ to obtain edge map:

$$E = \begin{bmatrix} 255 & 255 & 0 & 255 \\ 255 & 255 & 0 & 255 \\ 255 & 255 & 0 & 255 \\ 255 & 255 & 0 & 0 \end{bmatrix}$$

Problem 8: *5pt*

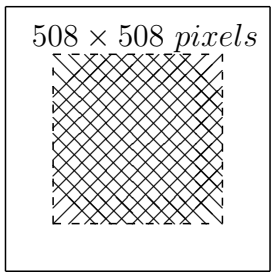
Solution:

a) If we do linear filtering directly: The size of resulting image would be $(512 + 5 - 1) \times (512 + 5 - 1) = 516 \times 516$.

b) If we implement the filtering use 512×512 point 2D DFT: The size of resulting image would be 512×512.

c) Since the origin of the filter is at the center, if we take the 516×516 image as reference, assume it starts with (0,0), the region that two image are equal would be within the region of $4 \leq x \leq 511$ and $4 \leq y \leq 511$.

On the other hand, if we take the 512×512 image as reference, assume it starts with (0,0), the region that two image are equal would be within the region of $2 \leq x \leq 509$ and $2 \leq y \leq 509$.



512×512 *pixels*

Or

516×516 *pixels*

As the figure shown above, the region that two image are equal would be a 508×508 region at the center of both images.

Problem 9: *20pt*

Solution:

a) Major steps for performing DFT domain filtering on an image F with a given filter mask in frequency domain H:

i) perform 2D-DFT on F to obtain $\tilde{F}$;

ii) multiply $\tilde{F}$ by H to obtain $\tilde{G}$;

iii) perform 2D-IDFT on $\tilde{G}$ to obtain desired filtered image G.

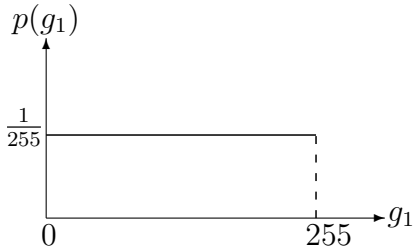
b) Matlab Programming:

```
close all; clear all;
F = imread('*_gray.jpg'); % Read image from file
[M,N] = size(F); % Determine image size
H = get_filter(M,N); % Get M*N filter
temp_F1 = zeros(M,N); temp_F2 = zeros(M,N);% save intermediate data

% apply 1D DFT for every row
for i = 1:M
    temp_F1(i,:) = fft(F(i,:));
end
% apply 1D DFT again for every column
for j = 1:N
    temp_F2(:,j) = fft(temp_F1(:,j));
end

%Filtering in Frequency domain by pointwise multiplication
Filtered_F = temp_F2.*H;

% Inverse FT, then take the real part
Filtered_F = real(ifft2(Filtered_F));
% Scale image into range (0-255) and convert to unsigned character ;
fmin=min(min(Filtered_F));
fmax=max(max(Filtered_F));
Filtered_F = uint8(255*(Filtered_F-fmin)/(fmax-fmin));
% Write filtered image into a jpg format file
imwrite(Filtered_F,'Filtered_image.jpg','jpg');
```

Problem 10: *Bonus 10pt*

Solution:

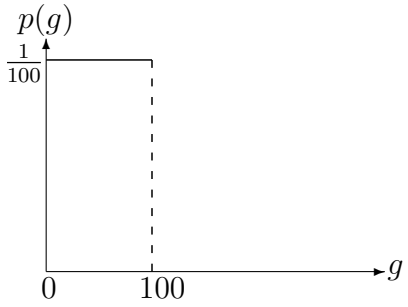
We can first do conventional histogram equalization to make the equalized image g_1 has a uniform distribution in (0,255) shown below. Then we can convert g_1 image to g image by a scaling.



Since $p(f) = ce^{-af}$ (where $c = a/(1 - e^{-255a})$), the transmformation from f to g_1 can be obtained by

$$g_1(f) = 255 \int_0^f p(f)df = 255\frac{c}{a}(1 - e^{-af})$$

where $c = a/(1 - e^{-255a})$.



Comparing the pdfs of g_1 and g shown above, g can be obtained from g_1 simply by linear stretching:

$$g = g_1\frac{100}{255} = 100\frac{c}{a}(1 - e^{-af}) = \frac{100}{1 - e^{-255a}}(1 - e^{-af})$$