ECE 638 Homework 5 Sample Solution

Question 1 (Weichen Xu)

1.

a.

Following is the threshold matrix to produce the dot profile function:

12	1	20	7	23	11	15	26
13	2	21	6	24	10	16	27
14	3	22	5	25	9	17	28
29	18	4	30	8	19	31	32
33	34	57	35	61	36	37	38
39	56	42	58	45	62	48	49
40	55	43	59	46	63	50	51
41	54	44	60	47	64	52	53

b.

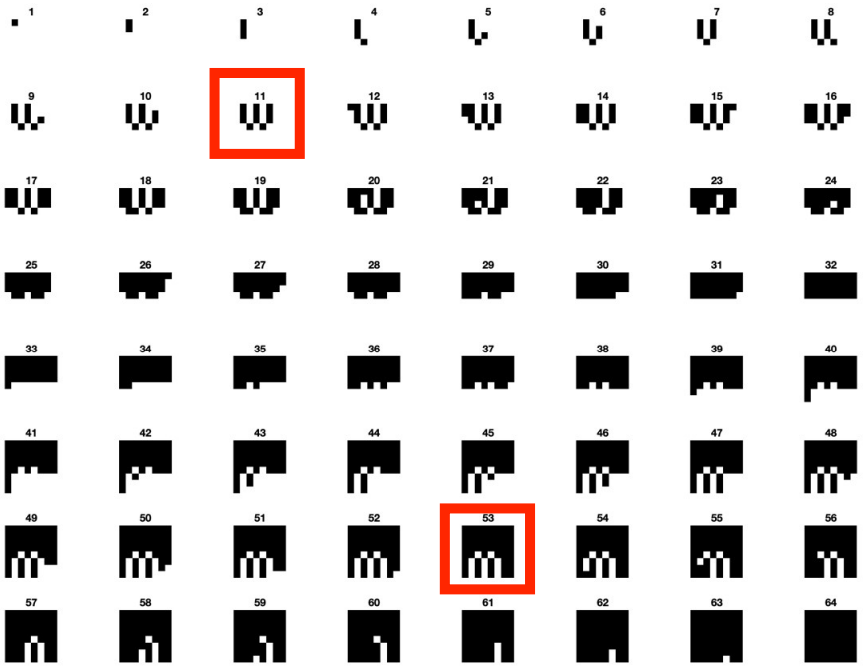


Fig 1: 65-level dot profile function plot

Question 2 (Kennedy F. Monaco)

a. Monochrome Screen MATLAB Code

```
function [result] = monochrome_screen(img, threshold)
result = zeros(length(img), length(img(1,:)));
dim = length(threshold);
for row = 1:length(img)
    t_row = mod(row, dim) + 1;
    for col = 1:length(img(1,:))
        t_col = mod(col, dim) + 1;
        if(img(row, col) >= threshold(t_row, t_col))
            result(row, col) = 255;
        end
    end
end
end
```

Figure 2.1: *Monochrome Screening Function*

b. Index and Threshold Matrices

3	1
2	4

31.8750	159.3750
95.6250	0

Figure 2.2: *2x2 Index Matrix (left) and associated Threshold Matrix (right)*

16	9	11	14
13	3	1	5
7	2	4	10
12	6	8	15

0	103.5938	71.7188	23.9063
39.8438	199.2188	231.0938	167.3438
135.4688	215.1563	183.2813	87.6563
55.7813	151.4063	119.5313	7.9688

Figure 2.3: *4x4 Index Matrix (left) and associated Threshold Matrix (right)*

64	53	49	36	40	45	57	61
60	44	25	17	18	26	41	54
48	32	16	5	9	13	27	50
39	24	12	1	2	6	19	33
35	23	8	4	3	10	20	37
52	31	15	11	7	14	28	46
56	43	30	22	21	29	42	58
63	59	47	38	34	51	55	62

0	41.8359	57.7734	109.5703	93.6328	73.7109	25.8984	9.9609
13.9453	77.6953	153.3984	185.2734	181.2891	149.4141	89.6484	37.8516
61.7578	125.5078	189.2578	233.0859	217.1484	201.2109	145.4297	53.7891
97.6172	157.3828	205.1953	249.0234	245.0391	229.1016	177.3047	121.5234
113.5547	161.3672	221.1328	237.0703	241.0547	213.1641	173.3203	105.5859
45.8203	129.4922	193.2422	209.1797	225.1172	197.2266	141.4453	69.7266
29.8828	81.6797	133.4766	165.3516	169.3359	137.4609	85.6641	21.9141
1.9922	17.9297	65.7422	101.6016	117.5391	49.8047	33.8672	5.9766

Figure 2.5: *8x8 Index Matrix (left) and associated Threshold Matrix (right)*

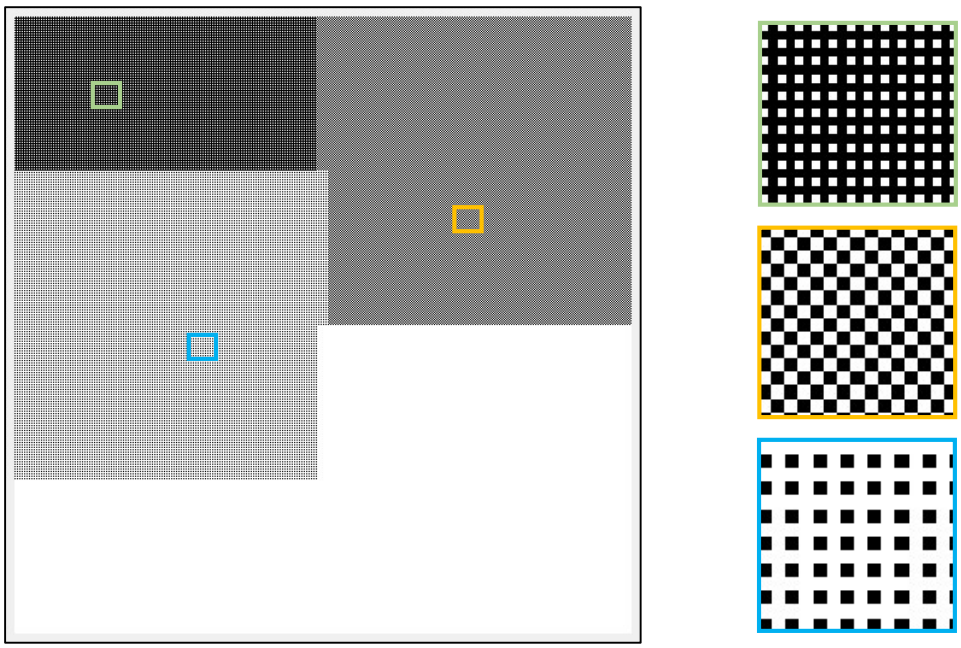


Figure 2.6: *Ramp Image applying 2x2 Threshold Matrix with sample patches to see halftoning texture*

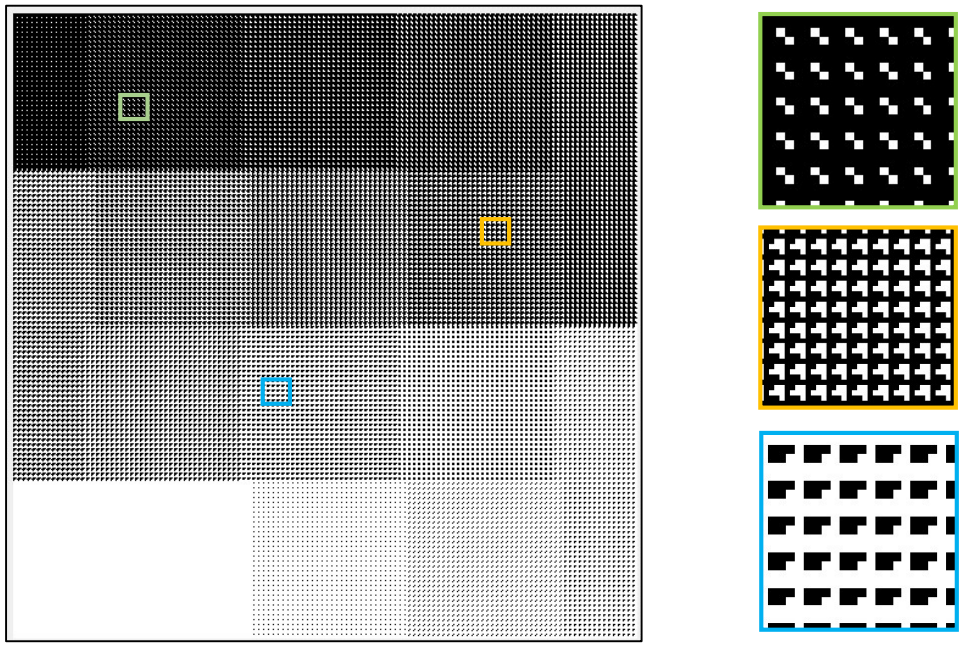


Figure 2.7: *Ramp Image applying 4x4 Threshold Matrix with sample patches to see halftoning texture*

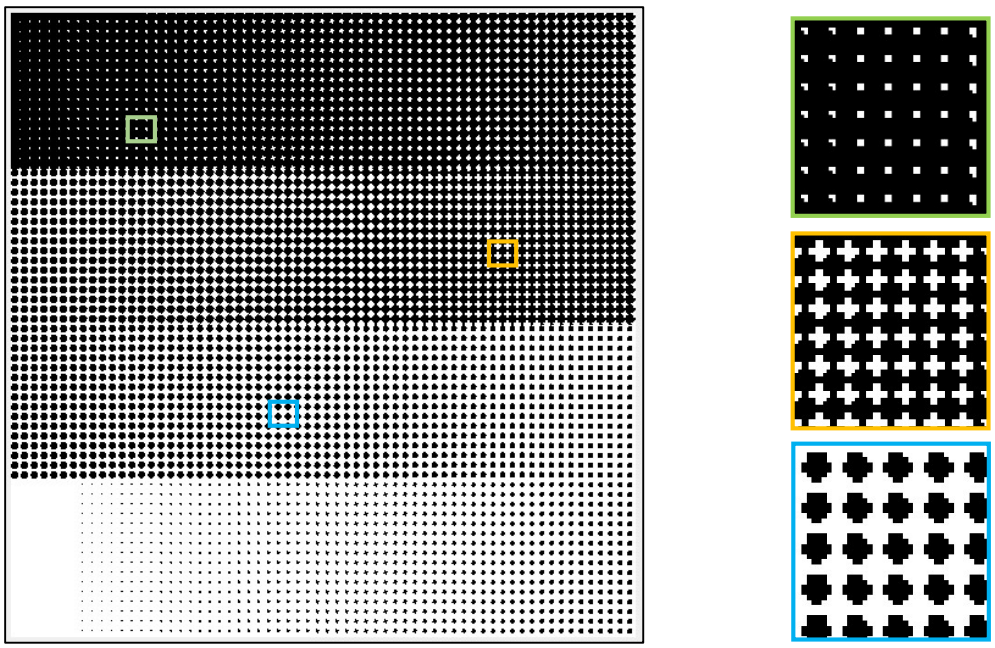


Figure 2.8: *Ramp Image applying 8x8 Threshold Matrix with sample patches to see halftoning texture*

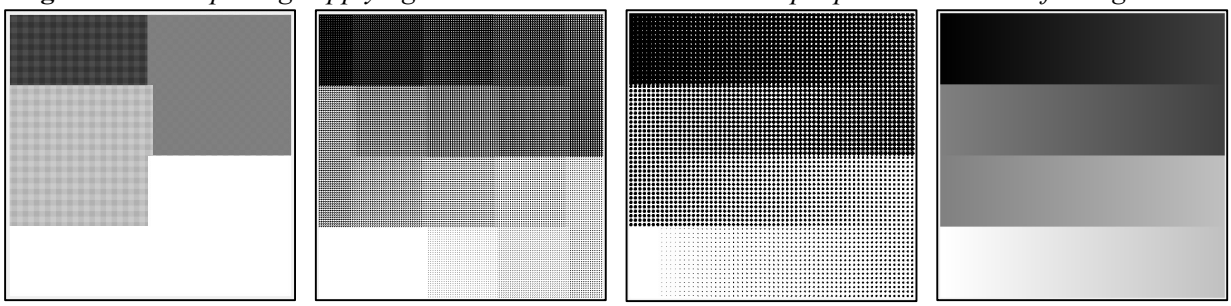


Figure 2.9: *Threshold Ramp Images and Original Ramp Image*

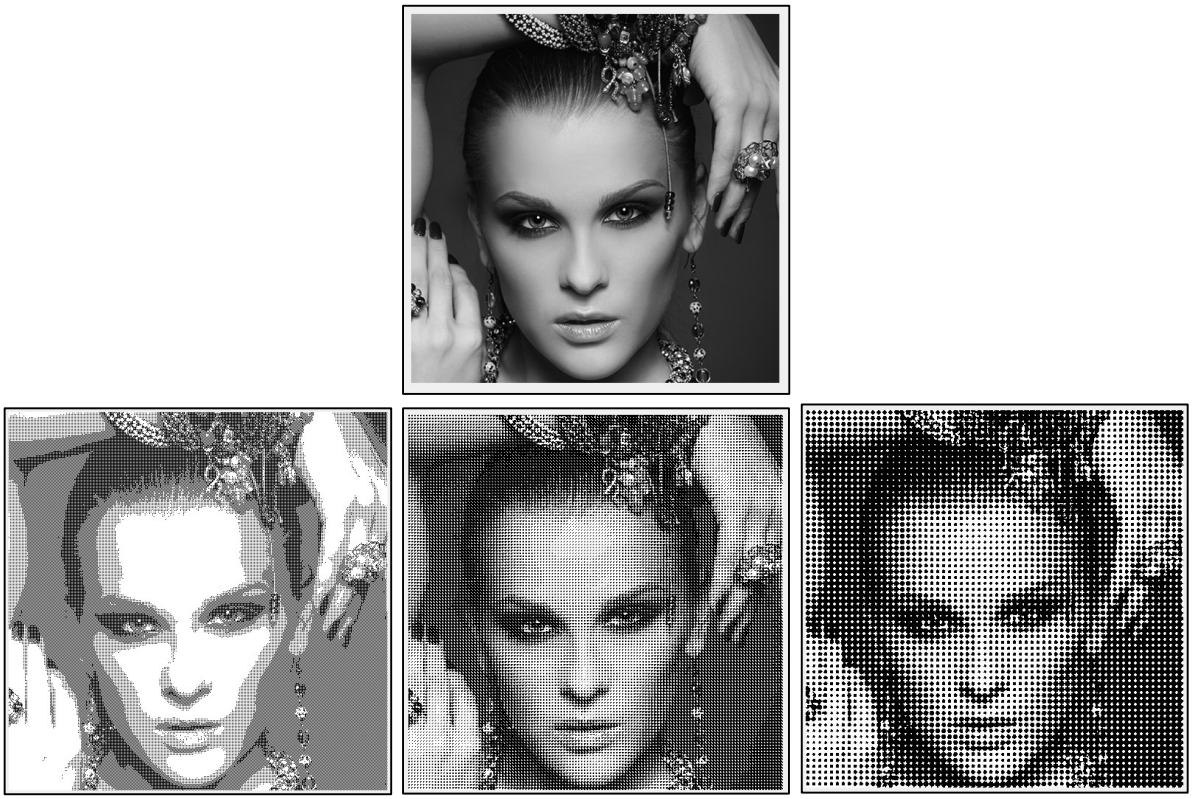


Figure 2.10: *Original Woman Image (top) and Woman Image applying 2x2 (left), 4x4 (center), and 8x8 (right) Threshold Matrices*

From the Black and White Woman Images, the 8x8 threshold matrix provides the best tone reproduction as the larger grey-tone palette has more flexibility when halftoning shadow and mid-tone areas in particular. The nails and background of the 8x8 threshold image are darker than those of the other images, which more closely resembles the original image. The 4x4 threshold matrix provides the best detail rendition, which is best exemplified in the woman's jewelry and ears. The woman's right ear is completely lost in the 2x2 threshold image and less visible in the 8x8 threshold image while it is clearly apparent in the 4x4 threshold image. The 8x8 threshold image loses a decent amount of fine detail as the period of the threshold matrix is too large to accommodate the finer features. The 2x2 threshold matrix does not provide enough grey-tones to capture some of the details such as the woman's right ear.

c. "K" Pattern Thresholding

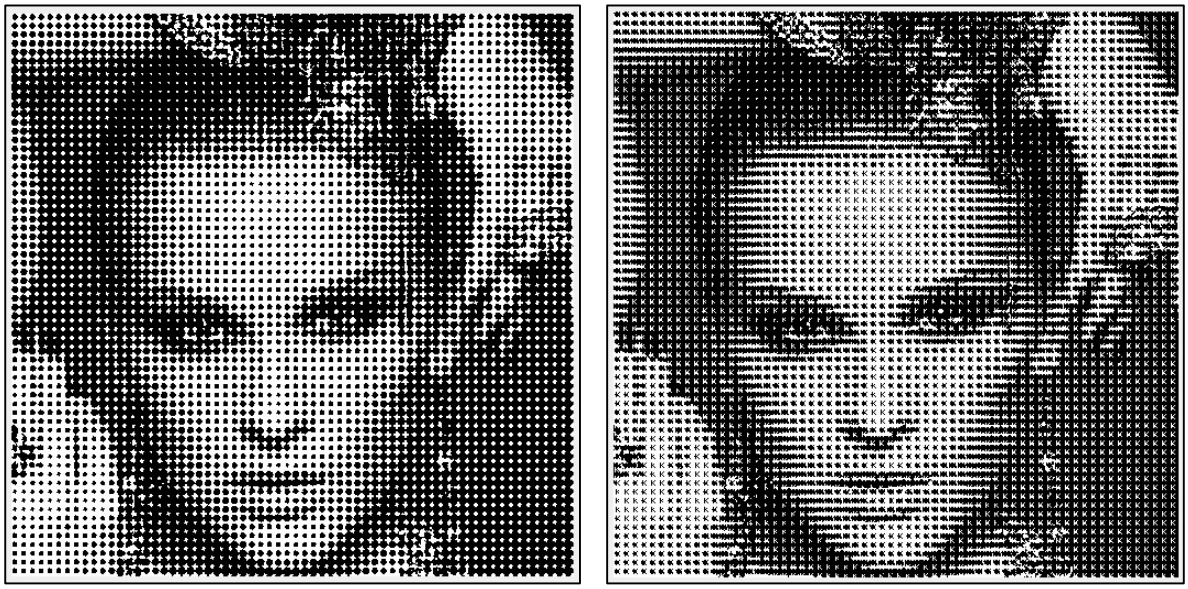
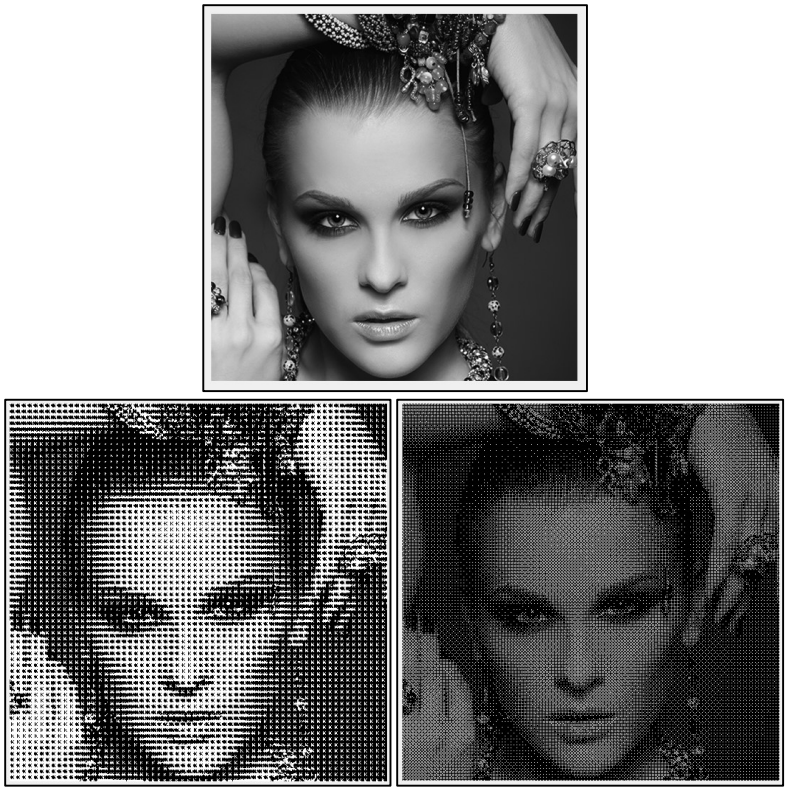


Figure 2.11: *8x8 Threshold Woman Image (left) and "K" Pattern Threshold Woman Image (right)*

The "K" threshold matrix applied image has significant artifacts that make the image lower quality than the 8x8 threshold applied image. The "K" pattern is clearly visible in the shadow and highlight regions while the mid-tone regions have horizontal line artifacts since the threshold matrix fills out horizontally, then vertically to ensure the fidelity of the "K" pattern in both the highlights and shadows. The "K" threshold matrix does, however, seem to have slightly better detail rendition in the woman's bracelets and certain facial features.



I don't understand why the Bayer thresholded image is darker than the original image.

Figure 2.12: *Original Image (top), "K" Threshold Image (left), and Bayer Threshold Image (right)*

The Bayer Threshold Image retains more of the original detail throughout the image (notably the woman's jewelry, eyes, and lips) than the "K" Threshold Image; however, the Bayer Threshold Image is significantly darker than the original image while the "K" Threshold Image is slightly lighter than the original but is closer in tone than the Bayer Threshold Image.

Question 3 (Tianqi Guo)

3. a. $p_i[x,y;a] = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} P_i[m,n;a]\,\delta(x-mR, y-nR)$

$= rep_{X,X} P_{i,X}[x,y;a]$

where $X = MR = 4R$

$P_{i,X}[x,y;a] = \sum_{m=0}^{3}\sum_{n=0}^{3} P_i[m,n;a]\,\delta(x-mR, y-nR)$

(i) for $i=A$, $a=\frac{4}{16}$

n↓				
0	0	0	0	0
1	0	1	1	0
2	0	1	1	0
3	0	0	0	0
m→	0	1	2	3

$P_{A,X}[x,y;a] = 1\cdot\delta(x-R,y-R)+1\cdot\delta(x-R,y-2R)$
$+1\cdot\delta(x-2R,y-R)+1\cdot\delta(x-2R,y-2R)$

$= rect(\frac{x-R}{R}, \frac{y-R}{R}) + rect(\frac{x-R}{R}, \frac{y-2R}{R}) + rect(\frac{x-2R}{R}, \frac{y-R}{R}) + rect(\frac{x-2R}{R}, \frac{y-2R}{R})$

$= rect(\frac{x-\frac{3}{2}R}{2R}, \frac{y-\frac{3}{2}R}{2R})$

$\Rightarrow P_A[x,y;\frac{4}{16}] = rep_{4R,4R}[rect(\frac{x-\frac{3}{2}R}{2R}, \frac{y-\frac{3}{2}R}{2R})]$

(ii) for $i=B$, $a=\frac{4}{16}$

n↓				
0	1	0	1	0
1	0	0	0	0
2	1	0	1	0
3	0	0	0	0
m→	0	1	2	3

$P_{B,X}[x,y;a] = 1\cdot\delta(x,y)+1\cdot\delta(x,y-2R)+1\cdot\delta(x-2R,y)+1\cdot\delta(x-2R,y-2R)$

$= rect(\frac{x}{R},\frac{y}{R}) + rect(\frac{x}{R},\frac{y-2R}{R}) + rect(\frac{x-2R}{R},\frac{y}{R}) + rect(\frac{x-2R}{R},\frac{y-2R}{R})$

$\Rightarrow P_B[x,y;\frac{4}{16}] = rep_{4R,4R}[P_{B,4R}[x,y;\frac{4}{16}]]$

$= rep_{2R,2R}[rect(\frac{x}{R},\frac{y}{R})]$

b. (i) $P_A[x,y;\frac{4}{16}] = rep_{4R,4R}[rect(\frac{x-\frac{3}{2}R}{2R}, \frac{y-\frac{3}{2}R}{2R})]$

$P_A[u,v;\frac{4}{16}] = \frac{1}{4R}\cdot\frac{1}{4R}\,comb_{\frac{1}{4R},\frac{1}{4R}}[|2R\cdot 2R|\,sinc(2Ru,2Rv)\,e^{-j2\pi\cdot\frac{3}{2}R(u+v)}]$

$= \frac{1}{4}\sum_m\sum_n sinc(2R\cdot m\cdot\frac{1}{4R}, 2R\cdot n\cdot\frac{1}{4R})\,e^{-j3\pi R(m\cdot\frac{1}{4R}+n\cdot\frac{1}{4R})}\,\delta(u-\frac{m}{4R}, v-\frac{n}{4R})$

$= \frac{1}{4}\sum_m\sum_n sinc(\frac{m}{2},\frac{n}{2})\,e^{-j\frac{3}{4}\pi(m+n)}\,\delta(u-\frac{m}{4R}, v-\frac{n}{4R})$

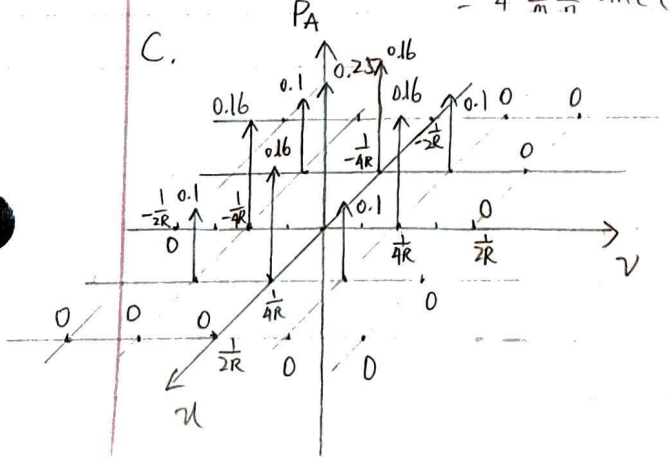
(ii) $P_B[x,y;\frac{4}{16}] = rep_{2R,2R}[rect(\frac{x}{R},\frac{y}{R})]$

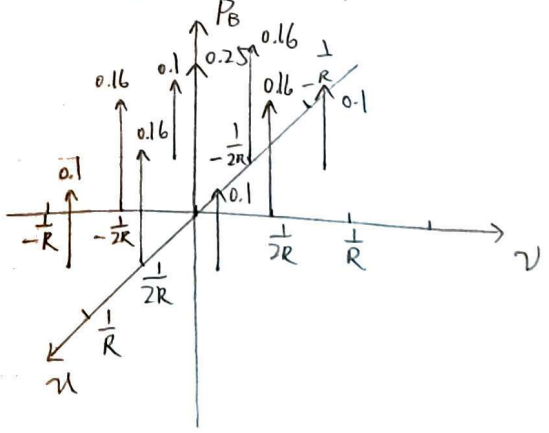
$P_B[u,v;\frac{4}{16}] = \frac{1}{2R}\cdot\frac{1}{2R}\,comb_{\frac{1}{2R},\frac{1}{2R}}[R^2\,sinc(Ru,Rv)]$

$= \frac{1}{4}\sum_m\sum_n sinc(R\cdot m\cdot\frac{1}{2R}, R\cdot n\cdot\frac{1}{2R})\,\delta(u-\frac{m}{2R}, v-\frac{n}{2R})$

$= \frac{1}{4}\sum_m\sum_n sinc(\frac{m}{2},\frac{n}{2})\,\delta(u-\frac{m}{2R}, v-\frac{n}{2R})$

C.





d. (i) For $i=A$, $a=\frac{6}{16}$

n ↓ \ m →	0	1	2	3
0	0	1	1	0
1	0	1	1	0
2	0	1	1	0
3	0	0	0	0

$$p_{AAR}[x,y;\tfrac{6}{16}] = 1\cdot d(x-R,y)+1\cdot d(x-2R,y)+p_{A,4R}[x,y,\tfrac{4}{16}]$$
$$= \text{rect}\left(\frac{x-R}{R},\frac{y}{R}\right)+\text{rect}\left(\frac{x-2R}{R},\frac{y}{R}\right)+\text{rect}\left(\frac{x-\frac{3}{2}R}{2R},\frac{y-\frac{3}{2}R}{2R}\right)$$
$$= \text{rect}\left(\frac{x-\frac{3}{2}R}{2R},\frac{y-2R}{3R}\right)$$
$$\Rightarrow p_A[x,y;\tfrac{6}{16}] = \text{rep}_{4RAR}[p_{A,4R}[x,y;\tfrac{6}{16}]]$$
$$= \text{rep}_{4RAR}\left[\text{rect}\left(\frac{x-\frac{3}{2}R}{2R},\frac{y-R}{3R}\right)\right]$$

(ii) For $i=B$, $a=\frac{6}{16}$

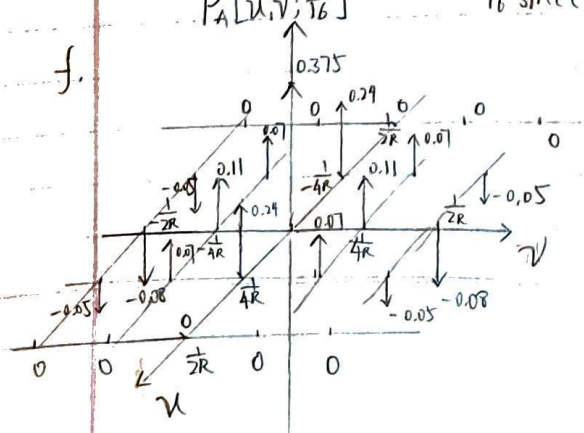
n ↓ \ m →	0	1	2	3
0	1	0	1	0
1	0	1	0	0
2	1	0	1	0
3	0	0	0	1

$$p_{B,4R}[x,y;\tfrac{6}{16}] = 1\cdot d(x-R,y-R)+1\cdot d(x-3R,y-3R)+p_{B,4R}[x,y;\tfrac{4}{16}]$$
$$= \text{rect}\left(\frac{x-R}{R},\frac{y-R}{R}\right)+\text{rect}\left(\frac{x-3R}{R},\frac{y-3R}{R}\right)+p_{B,4R}[x,y;\tfrac{4}{16}]$$
$$\Rightarrow p_B[x,y;\tfrac{6}{16}] = \text{rep}_{4RAR}[p_{BAR}[x,y;\tfrac{6}{16}]]$$
$$= \text{rep}_{4R,AR}\left[\text{rect}\left(\frac{x-R}{R},\frac{y-R}{R}\right)+\text{rect}\left(\frac{x-3R}{R},\frac{y-3R}{R}\right)\right]$$
$$+\text{rep}_{2R,2R}\left[\text{rect}\left(\frac{x}{R},\frac{y}{R}\right)\right]$$

e. (i) $P_A[u,v;\frac{6}{16}] = \text{CSFT}\{p_A(x,y;\frac{6}{16})\}$
$$= \frac{1}{4R}\cdot\frac{1}{4R}\,\text{comb}_{\frac{1}{4R},\frac{1}{4R}}\left[2R\cdot 3R\cdot\text{sinc}(2Ru,3Rv)\,e^{-j2\pi\left(\frac{3}{2}Ru+Rv\right)}\right]$$
$$= \frac{3}{8}\sum_m\sum_n \text{sinc}\left(2R\cdot m\cdot\frac{1}{4R},3R\cdot n\cdot\frac{1}{4R}\right)e^{-jR\pi\left(3\cdot\frac{m}{4R}+2\cdot\frac{n}{4R}\right)}\,\delta\left(u-\frac{m}{4R},v-\frac{n}{4R}\right)$$
$$= \frac{3}{8}\sum_m\sum_n \text{sinc}\left(\frac{m}{2},\frac{3}{4}n\right)e^{-j\pi\left(\frac{3}{4}m+\frac{1}{2}n\right)}\,\delta\left(u-\frac{m}{4R},v-\frac{n}{4R}\right)$$

(ii) $P_B[u,v;\frac{6}{16}] = \text{CSFT}\{p_B(x,y;\frac{6}{16})\}$
$$= \text{CSFT}\left\{\text{rep}_{2R,2R}\left[\text{rect}\left(\frac{x}{R},\frac{y}{R}\right)\right]+\text{rep}_{4RAR}\left[\text{rect}\left(\frac{x-R}{R},\frac{y-R}{R}\right)+\text{rect}\left(\frac{x-3R}{R},\frac{y-3R}{R}\right)\right]\right\}$$
$$= P_B[u,v;\tfrac{4}{16}]+\frac{1}{4R}\cdot\frac{1}{4R}\,\text{comb}_{\frac{1}{4R},\frac{1}{4R}}\left[R^2\text{sinc}(Ru,Rv)e^{-j2\pi R(u+v)}+R^2\text{sinc}(Ru,Rv)e^{-j2\pi(3u+3v)}\right]$$
$$= P_B[u,v;\tfrac{4}{16}]+\frac{1}{16}\,\text{comb}_{\frac{1}{4R},\frac{1}{4R}}\left[\text{sinc}(Ru,Rv)\left(e^{-j2\pi R(u+v)}+e^{-j6\pi R(u+v)}\right)\right]$$
$$= P_B[u,v;\tfrac{4}{16}]+\sum_m\sum_n\frac{1}{16}\text{sinc}\left(\frac{m}{4},\frac{n}{4}\right)\left(e^{-j\frac{\pi}{2}(m+n)}+e^{-j\frac{3}{2}\pi(m+n)}\right)\delta\left(u-\frac{m}{4R},v-\frac{m}{4R}\right)$$
$$= \sum_m\sum_n\left\{\frac{1}{4}\text{sinc}\left(\frac{m}{2},\frac{n}{2}\right)\delta\left(u-\frac{m}{2R},v-\frac{n}{2R}\right)+\right.$$
$$\left.\frac{1}{16}\text{sinc}\left(\frac{m}{4},\frac{n}{4}\right)\delta\left(u-\frac{m}{4R},v-\frac{n}{4R}\right)\left(e^{-\frac{j\pi}{2}(m+n)}+e^{-\frac{3j\pi}{2}(m+n)}\right)\right\}$$

f. $P_A[u,v;\frac{6}{16}]$



$P_B[u,v;\frac{6}{16}]$

see next page

In [1]:

```
# Problem 3. f.g.h.
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.ticker import LinearLocator
```

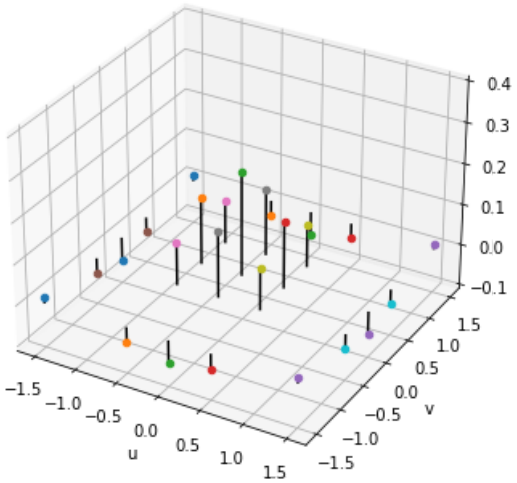
In [2]:

```
def draw_spectrum(ax, XX, YY, ZZ, is_slice):
    for i in range(len(XX)):
        if abs(ZZ[i])<0.01:continue
        if is_slice and XX[i] != 0: continue
        ax.plot3D([XX[i],XX[i]], [YY[i],YY[i]], [0, ZZ[i]],'k')
        ax.scatter3D(XX[i], YY[i], ZZ[i],'ro')
        if is_slice: ax.text(XX[i], YY[i], ZZ[i], '%0.3f'%(ZZ[i]))
    ax.set_xlabel('u')
    ax.set_ylabel('v')
    ax.set_zlim([-0.1,0.4])
    if is_slice: ax.view_init(elev=0, azim=0)
```

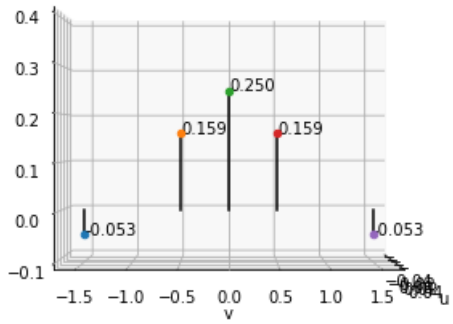
In [3]:

```
R = 1
step = 1/(4*R)
n = int(4*R/step)+1
x = np.linspace(-2*R, 2*R, n)
y = np.linspace(-2*R, 2*R, n)
X, Y = np.meshgrid(x, y)
Z_1 = 0.250*np.sinc(R*X)*np.sinc(R*Y)
for i in range(n):
    for j in range(n):
        if i%2!=0 or j%2!=0:
            Z_1[i][j] = 0
Z_2 = 0.125*np.sinc(R*X)*np.sinc(R*Y)
Z = Z_1 + Z_2
fig = plt.figure(figsize=[15,20])
XX = X.flatten()
YY = Y.flatten()
ZZ = Z_1.flatten()
ax = fig.add_subplot(321, projection='3d')
draw_spectrum(ax, XX, YY, ZZ, False)
ax.set_title('P_B from previous')
ax = fig.add_subplot(322, projection='3d')
draw_spectrum(ax, XX, YY, ZZ, True)
ax.set_title('P_B from previous: mid-plane')
ZZ = Z_2.flatten()
ax = fig.add_subplot(323, projection='3d')
draw_spectrum(ax, XX, YY, ZZ, False)
ax.set_title('Additional P_B terms')
ax = fig.add_subplot(324, projection='3d')
draw_spectrum(ax, XX, YY, ZZ, True)
ax.set_title('Additional P_B terms: mid-plane')
ZZ = Z.flatten()
ax = fig.add_subplot(325, projection='3d')
draw_spectrum(ax, XX, YY, ZZ, False)
ax.set_title('Total P_B')
ax = fig.add_subplot(326, projection='3d')
draw_spectrum(ax, XX, YY, ZZ, True)
ax.set_title('Total P_B: mid-plane')
plt.show()
```

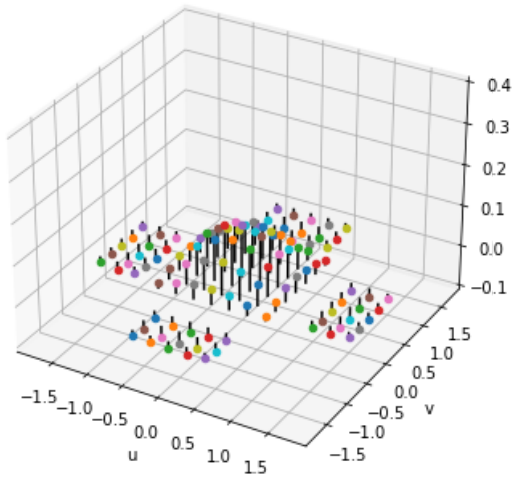
P_B from previous



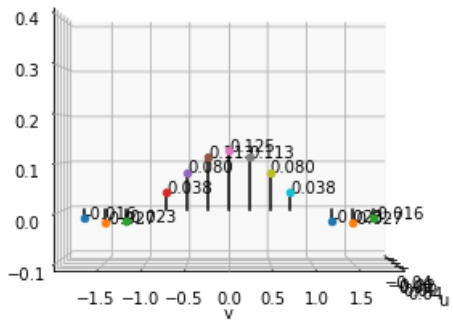
P_B from previous: mid-plane



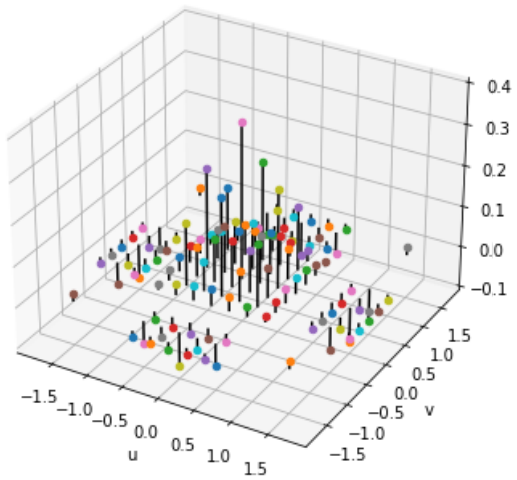
Additional P_B terms



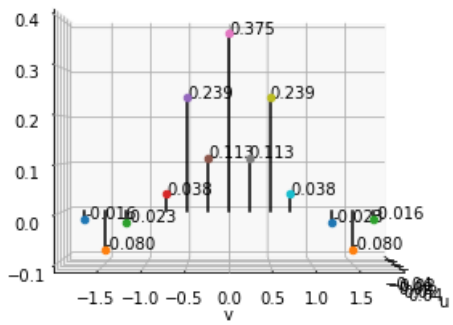
Additional P_B terms: mid-plane



Total P_B



Total P_B: mid-plane



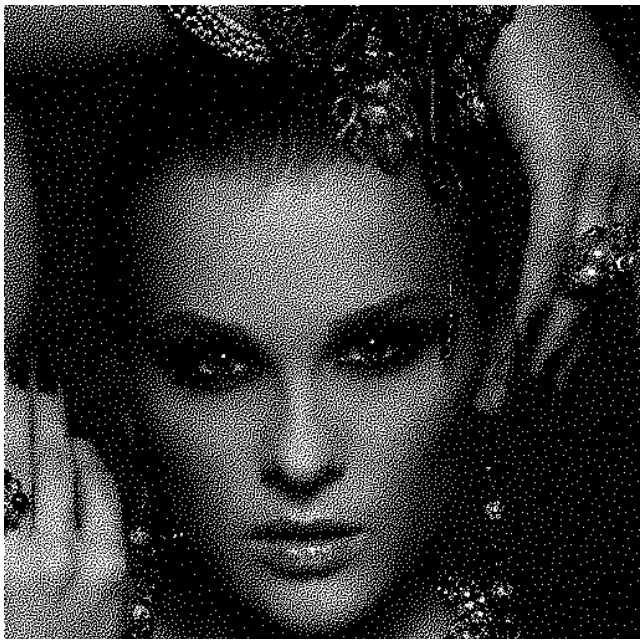
In [ ]:

In [4]:
```
# Problem 3.g.
# Similarity: P_A[4/16] and P_B[4/16] / P_A[6/16] and P_B[6/16] have the same DC gains,
#             meaning the overall image brightness after halftoning is the same
#             for each absorptance level
# Difference: As Dot Profile A is clustered and Dot Profile B is dispersed
#             in frequence domain we see P_B is more spread than P_A for each absorptan
#             as a result P_B halftoning has better fine details preserved
# Dot Profile A: as absorptance level increases from 4/16 to 6/16,
#                DC gain of P_A changes which reflects the brightness level
#                also P_A is no longer axis symmetric as frequence response in u and v
# Dot Profile B: as absorptance level increases from 4/16 to 6/16,
#                DC gain of P_A changes which reflects the brightness level
#                also the envolope of P_B is no longer a simple sinc function
#                as the sinc is now sampled at two different frequencies and superposit
```

In [5]:
```
# Problem 3.h.
# with a cut-off frequency just below 1/(2R)
# P_A[4/16] and P_B[4/16] should now have very similar overall brightness level
# however, while P_B[4/16] samples at 1/(2R) while P_A[4/16] samples at 1/(4R)
# the human observer will no longer see the dot effect from P_B[4/16],
# and it looks almost like continuously-toned image
# while dot effect is still quite visible for P_A[4/16]
# overall P_B gives better viewing quality than P_A

# P_A[6/16] and P_B[6/16] have similar comparisons
# but since P_B[6/16] now has additional terms sampled at 1/(4R)
# some dot effects are visible by human viewers
# and as a result it does not appear like continuously-toned image
```

Question 4 (Qiyue Liang)



d. Random white noise halftoned woman image

The quality of image halftoned by random white noise is very messy, the quality of the image halftoned by void and cluster looks much better. The void and cluster make sure the dots are oriented in a good manner, while the white noise make the image noisy.

```
%This is for ECE638 HW5 Q4 void_and_cluster
%Author: Qiyue Liang
%Date: Oct 30, 2021
close all
clear all
clc

%initial binary pattern generator
M = 128;
N = 128;
init = randi([0,1],M,N); %if more than half is 1 we should switch 0 and 1
if sum(init(:)) > (M*N/2)
    init = double(~init);
end


r1 = 1; r2 = 2; c1 = 1; c2 = 2; %initialize search points
stepcount = 0; %if step = 0 do nothing
while ~(r1 == r2 && c1 == c2) %they equal when removing c generate max v
    if stepcount ~= 0
        init(r2, c2) = 1;
    end
    [r1, c1] = findLargest('cluster', init);
    stepcount = stepcount + 1;
    init(r1, c1) = 0; %if removing the point in tightest cluster will
    [r2, c2] = findLargest('void', init); %generate the largest point
end
init(r1, c1) = 1; %put the point in the tightest cluster back

%phase1
phase1 = init;
dither = zeros(M,N);
rank = sum(phase1(:)) - 1;
while rank>=0
    [r,c] = findLargest('cluster', phase1);
    phase1(r,c) = 0;
    dither(r,c) = rank;
    rank = rank - 1;
end

%phase2
phase2 = init;
rank = sum(phase2(:));
while rank < (M*N/2)
    [r,c] = findLargest('void', phase2);
    phase2(r,c) = 1;
    dither(r,c) = rank;
    rank = rank+1;
end

%phase3
phase3 = ~phase2; %reverse 0 and 1
while rank < (M*N)
    [r,c] = findLargest('cluster', phase3);
    phase3(r,c) = 0;
```

```
    dither(r,c) = rank;
    rank = rank + 1;
end

%normalization
rankprime = floor((256-1)/(2-1)/(M*N).*(dither+1/2));

%start filtering the images
img = double(imread('woman_bw.tiff'));
[h,w] = size(img);
gamma = 2.2;
ungamma = 255.*(img./255).^gamma;

halfvac = zeros(size(img));
for j = 1:M:h
    for k = 1:N:w
        halfvac(j:j+M-1, k:k+N-1) = (ungamma(j:j+M-1, k:k+N-1) > rankprime);
    end
end

imwrite(halfvac, 'void_and_cluster.tiff');

%generate random white noise
wn = randi([0 255], M, N);
halfwn = zeros(size(img));
for j = 1:M:h
    for k = 1:N:w
        halfwn(j:j+M-1, k:k+N-1) = (ungamma(j:j+M-1, k:k+N-1) > wn);
    end
end

imwrite(halfwn, 'white_noised.tiff');

mse_vac = sum(sum((ungamma - halfvac).^2))/(h*w);
mse_wn = sum(sum((ungamma - halfwn).^2))/(h*w);


function [r,c] = findLargest(flag, arr)
[M,N] = size(arr);

%generate lookup table
% MN33 = repmat(arr, 3); %since x from 0 to M, neiber of x from -0.5M to 1.5M

MN33 = ones(3*M,3*N);
for r=1:M:3*M
    for c=1:N:3*N
        MN33(r:r+M-1,c:c+N-1) = arr;
    end
end
lookup = MN33(floor(0.5*M) + 1 : floor(2.5*M), floor(0.5*N) + 1 :
floor(2.5*N));%scale up from 0.5M to 2.5M or 0 to 3M
%imfilter does a convolution like filtering on the image
DA = imfilter(lookup, fspecial('gaussian', [M N], 1.5)); %we only want one
size of M,N, when we crop first time from 0.5 to 2.5
DA = DA(floor(0.5*M) + 1 : floor(1.5*M), floor(0.5*N)+1 : floor(1.5*M));
```

```
if strcmp(flag, 'cluster')
    DA(arr == 0) = -1;
    [maxval, maxpos] = max(DA(:)); %for every point
    [r,c] = ind2sub(size(DA),maxpos);
end

if strcmp(flag, 'void')
    DA(arr == 1) = M*N;
    [minval, minpos] = min(DA(:));
    [r,c] = ind2sub(size(DA),minpos);
end
end
```

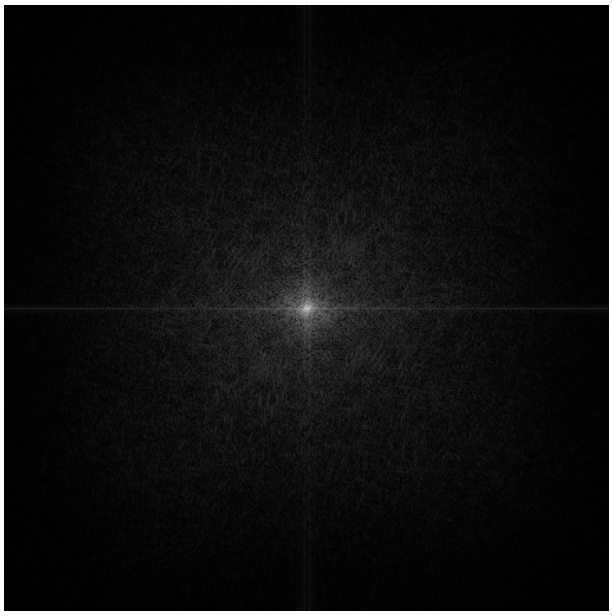
Question 5 (Fan Bu)

b

I choose $\kappa = 2000$ to generate the following images.

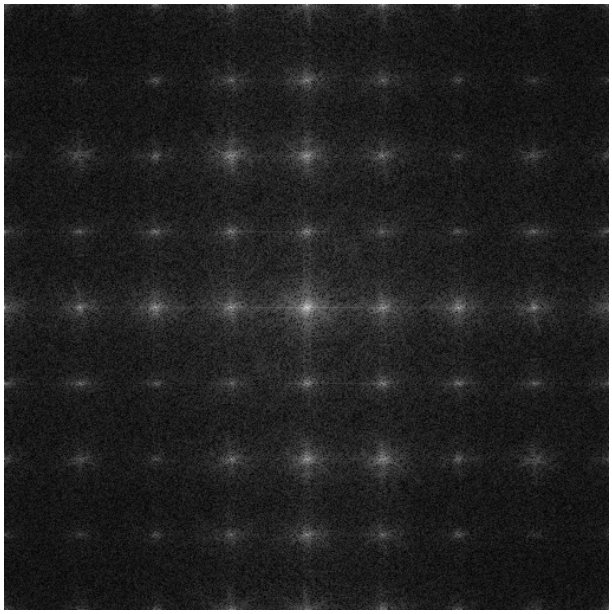
i

The 2D-DFT of the *woman_bw* image is shown below:



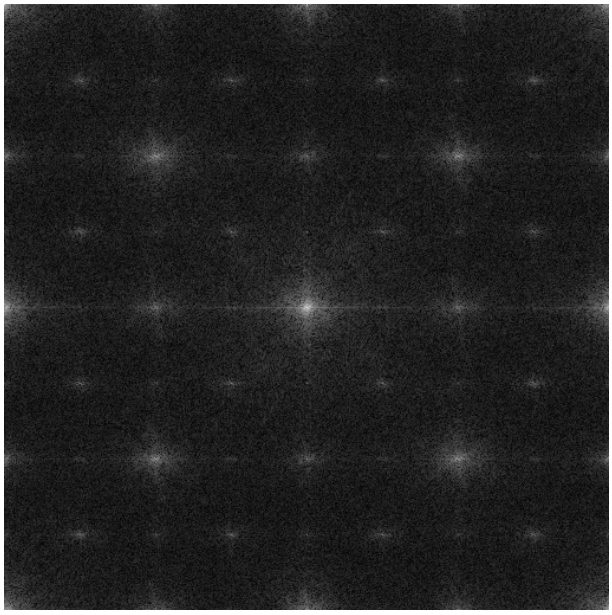
ii

The 2D-DFT of the clustered-dot periodic halftone version of the *woman_bw* image using my 8×8 threshold array is shown below:



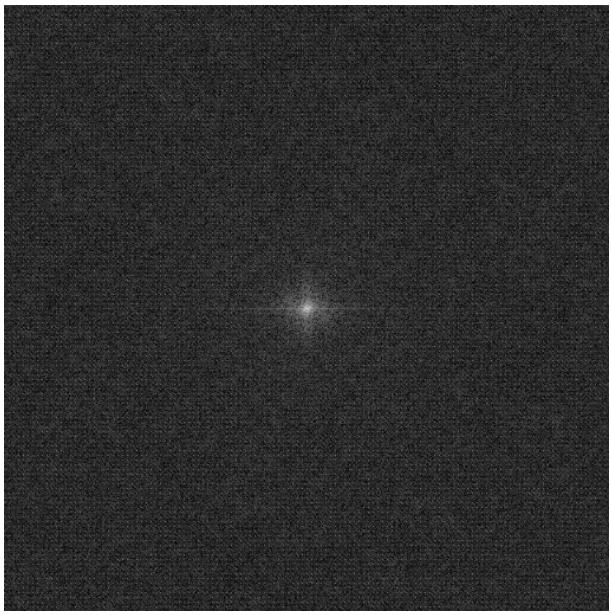
iii

The 2D-DFT of the dispersed-dot periodic halftone version of the *woman_bw* image using the 8×8 Bayer threshold array is shown below:



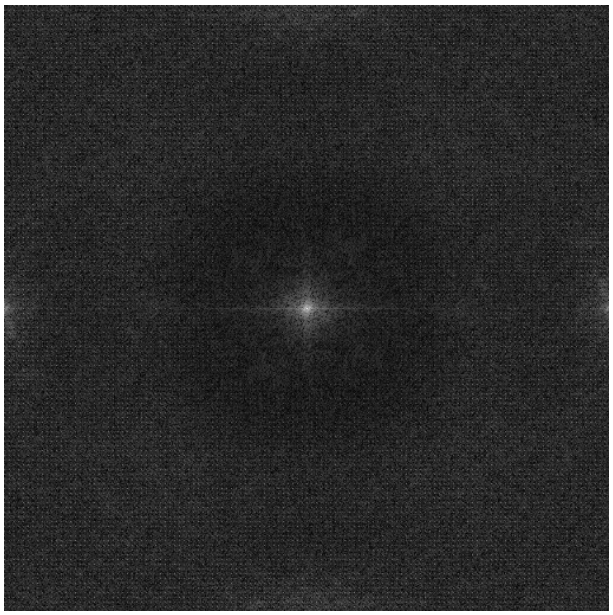
iv

The 2D-DFT of the *woman_bw* image thresholding with the white noise is shown below:



v

The 2D-DFT of the halftone version of the *woman_bw* image using the 128×128 void-and-cluster is shown below:



A larger value for \kappa might better show the suppression of energy near the original of the spectrum.

c

In question (b)(i), in the 2D-DFT of the original *woman_bw* image, there is not much special about it.

In question (b)(ii), the 2D-DFT of the clustered-dot periodic halftone version of the *woman_bw* image using my 8×8 threshold array contains checkerboard patterns. They are created when halftoned the image by simple spiral clustered-dot threshold arrays.

In question (b)(iii), the 2D-DFT of the dispersed-dot periodic halftone version using the 8×8 Bayer threshold array also shows the checkerboard patterns. The distribution of the peaks shows two main periodic frequencies, which matches our deduction in problem 2.

In question (b)(iv), the 2D-DFT of the *woman_bw* image thresholding with the white noise look similar to the 2D-DFT of the original *woman_bw* image. However, the white noise amplifies the signals in high frequencies.

In question (b)(v), the 2D-DFT of the halftone version of the *woman_bw* image using the 128×128 void-and-cluster contains an energy lattice around the dominant component. We observe that the radius of the energy lattice is roughly 125 pixels. Hence an estimation of the average principal frequency for the entire halftone image is: $\frac{1}{125} = 0.008$ cycles/pixel.

This should be \mu = `125/512 = 0.244 cycles/pixel

```
    end
end
end

% Code for HW05 Question4, by Fan Bu, Nov. 2021.
clear all; close all; clc;

%% (b)
%Initialize Binary Pattern
M = 128;
input_BP_size = int32(round(M*M*0.4));
BP = zeros(M,M);
for m = 1:input_BP_size
    i = ceil(rand*M);
    j = ceil(rand*M);
    BP(i,j) = 1;
end
input_BP_img = uint8((1-BP)*255);
% imwrite(input_BP_img,'HW05_4_b_input_BP.png');

% Generate Binary Pattern
while(true)
    % Find the tightest cluster
    DA = (imgaussfilt(BP,1.5,'Padding','circular')).*BP;
    [max_tc,idx_tc]=max(DA(:));
    [imax_tc,jmax_tc]=ind2sub(size(DA),idx_tc);
    % move "1" away from the tightest cluster
    BP(imax_tc,jmax_tc)=0;

    % Find the largest void
    BP_white = 1-BP;
    da2 = (imgaussfilt(BP_white,1.5,'Padding','circular')).*BP_white;
    [max_lv,idx_lv]=max(da2(:));
    [imax_lv,jmax_lv]=ind2sub(size(da2),idx_lv);
    if (imax_lv==imax_tc)&&(jmax_lv==jmax_tc)
        BP(imax_tc,jmax_tc)=1;
        break;
    end

    % insert "1"  to the largest void
    BP(imax_lv,jmax_lv)=1;
```

```
    %input_BP_img_debug = uint8((1-BP)*255);
    %imwrite(input_BP_img_debug,'HW05_4_b_input_BP_debug.png');
end
BP_img = uint8((1-BP)*255);
imwrite(BP_img,'HW05_4_b_BP.png');

%Dither Array Generator
BP_rank = sum(BP(:)==1)-1;
void_cluster_M = ones(M,M)*(-1);
% phase 1
while (BP_rank>=0)
    DA_1 = (imgaussfilt(BP,1.5,'Padding','circular')).*BP.*(void_cluster_M==-1);
    [max_tc_1,idx_tc_1]=max(DA_1(:));
    [imax_tc_1,jmax_tc_1]=ind2sub(size(DA_1),idx_tc_1);
    BP(imax_tc_1,jmax_tc_1) = 0;
    void_cluster_M(imax_tc_1,jmax_tc_1) = BP_rank;
    BP_rank = BP_rank - 1;
end
% phase 2
BP_rank = input_BP_size;
while (BP_rank<M*M/2)
    BP_white = 1-BP;
    DA_2 = (imgaussfilt(BP_white,1.5,'Padding','circular')).*BP_white.*(void_cluster_M==-1);
    [max_lv_2,idx_lv_2]=max(DA_2(:));
    [imax_lv_2,jmax_lv_2]=ind2sub(size(DA_2),idx_lv_2);
    BP(imax_lv_2,jmax_lv_2) = 1;
    void_cluster_M(imax_lv_2,jmax_lv_2) = BP_rank;
    BP_rank = BP_rank + 1;
end
% phase 3
while (BP_rank<M*M)
    BP_white = 1-BP;
    da_p3 = (imgaussfilt(BP_white,1.5,'Padding','circular')).*BP_white.*(void_cluster_M==-1);
    [maxv_p3,idxv_p3]=max(da_p3(:));
    [imaxv_p3,jmaxv_p3] = ind2sub(size(da_p3),idxv_p3);
    BP(imaxv_p3,jmaxv_p3) = 1;
    void_cluster_M(imaxv_p3,jmaxv_p3) = BP_rank;
    BP_rank = BP_rank + 1;
end
BP_img = uint8((1-BP)*255);
```

```
imwrite(BP_img,'HW05_4_b.png');

%% (c)
% halftone the woman_bw image with Void and Cluster
img_woman_bw = double(imread('woman_bw.tif'));
[img_h, img_w] = size(img_woman_bw);
img_woman_bw_VC = monochrome_screen(img_woman_bw, void_cluster_M);
imwrite(img_woman_bw_VC,'HW05_4_c.png');

%% (d)
% halftone the woman_bw image with white noise
M = 128;
M_wn = zeros(M,M);
for i = 1:M
    for j = 1:M
        M_wn(i,j) = ceil(rand*M*M);
    end
end
img_woman_bw_wn = monochrome_screen(img_woman_bw, M_wn);
imwrite(img_woman_bw_wn,'HW05_4_d.png');

function output_img = monochrome_screen(input_img,threshold_matrix)
[img_h,img_w] = size(input_img);
[m,~] = size(threshold_matrix);
output_img = zeros(img_h,img_w);
thr_m = 255.*(threshold_matrix+0.5)./m^2;
for i = 1:m:img_h
    for j = 1:m:img_w
        img_kernel = 255.*ones(m,m);
        img_temp = input_img(i:i+m-1,j:j+m-1);
        img_kernel(img_temp<thr_m) = 0;
        output_img(i:i+m-1,j:j+m-1)= img_kernel;
    end
end
end

% Code for HW05 Question5, by Fan Bu, Nov. 2021.
clear all; close all; clc;

%% (b)
K = 2000;
```

```
img_woman_bw_1=imread('woman_bw.tif');
H_out1 = img_2D_dft(img_woman_bw_1,K);
imwrite(uint8(H_out1),'HW05_5_b_1.png');

img_woman_bw_2 = imread('HW05_2_c_woman_8x8.png');
H_out2 = img_2D_dft(img_woman_bw_2,K);
imwrite(uint8(H_out2),'HW05_5_b_2.png');

img_woman_bw_3 = imread('HW05_2_d_woman_8x8.png');
H_out3 = img_2D_dft(img_woman_bw_3,K);
imwrite(uint8(H_out3),'HW05_5_b_3.png');

img_woman_bw_4 = imread('HW05_4_d.png');
H_out4 = img_2D_dft(img_woman_bw_4,K);
imwrite(uint8(H_out4),'HW05_5_b_4.png');

img_woman_bw_5 = imread('HW05_4_c.png');
H_out5 = img_2D_dft(img_woman_bw_5,K);
imwrite(uint8(H_out5),'HW05_5_b_5.png');



function H_out = img_2D_dft(input_img,k_compress)
H_in = abs(fftshift(fft2(input_img)));
%compression equation
numerator = log(k_compress*H_in./max(H_in(:))+1);
denominator = max(max(log(k_compress*H_in./max(H_in(:))+1)));
H_out = 255*numerator/denominator;
end
```