Lab 8 Report

Ruijie Song

Apr.16.2021

3.1

Figure 1. original image

Figure 2. result of thresholding

RMSE = 87.3933165438293

Fidelity = 77.3371491724335

4.2 Exercise: Ordered Dithering

1.

1	2
3	0

Figure 3. 2x2 Bayer index matrices

5	9	6	10
13	1	14	2
7	11	4	8
15	3	12	0

Figure 4. 4x4 Bayer index matrices

21	37	25	41	22	38	26	42
53	5	57	9	54	6	58	10
29	45	17	33	30	46	18	34
61	13	49	1	62	14	50	2
23	39	27	43	20	36	24	40
55	7	59	11	52	4	56	8
31	47	19	35	28	44	16	32
63	15	51	3	60	12	48	0

Figure 5. 8x8 Bayer index matrices

2.

Figure 6. halftoned image produced by 2x2 matix

Figure 7. halftoned image produced by 4x4 matix

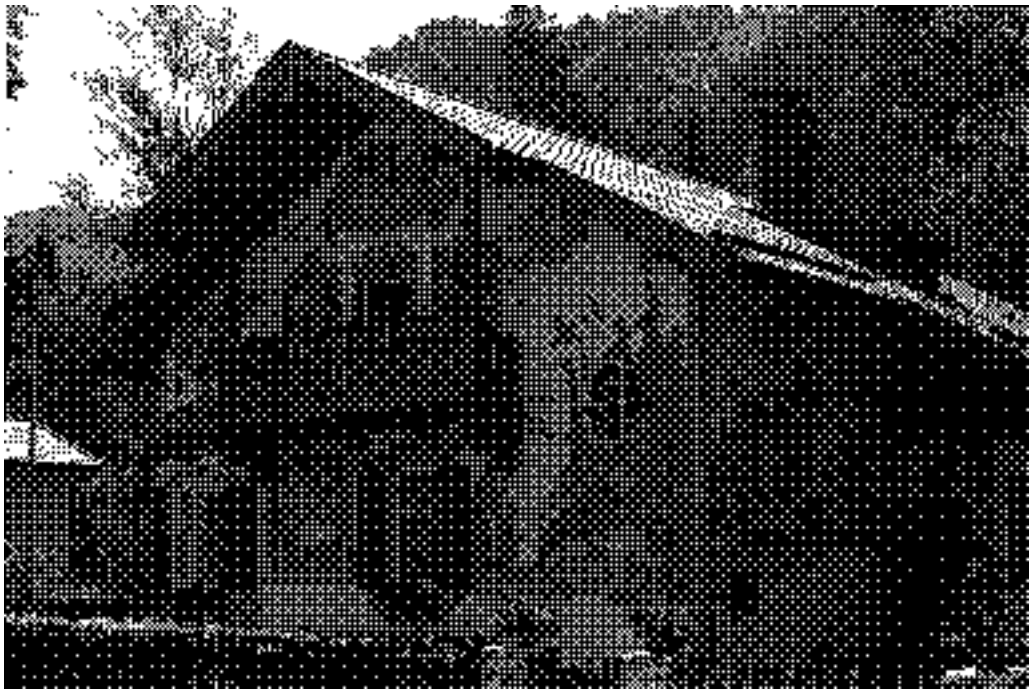

Figure 8. halftoned image produced by 8x8 matix

3.

RMSE_2x2 = 97.6689721921400
Fid_2x2 = 50.056947960301500
RMSE_4x4 = 1.010069220156947e+02
Fid_4x4 = 16.558342510690416
RMSE_8x8 = 1.009145296239608e+02
Fid_8x8 = 14.691773433816397

5.1 Exercise: Error Diffusion

2.

3.

RMSE = 99.192402741527390

Fidelity = 14.140898102731818

4.

	RMSE	fidelity
simple thresholding	87.3933165438293	77.3371491724335
2x2	97.6689721921400	50.056947960301500
4x4	1.010069220156947e+02	16.558342510690416
8x8	1.009145296239608e+02	14.691773433816397
error diffusion	99.192402741527390	14.140898102731818

From the results, we can observe that RMSE does not change much for different methods, but fidelity decreases for different methods. When the quality is better, the fidelity is smaller.

```
clear all
%{
%% 3.1
% 1
house = imread('house.tif');
[rowh,colh] = size(house);
T = 127; % threshold
binary = zeros(rowh,colh);
for i = 1:rowh
    for j = 1:colh
        if house(i,j) > T
            binary(i,j) = 255;
        else
            binary(i,j) = 0;
        end
    end
end

figure()
imshow(binary)

% 2
house = double(house);
square = (binary-house).^2;
RMSE = sqrt(1/(rowh*colh) * sum(square(:)));
% 3
fid = fidelity(house,binary);
%}
%{
%% 4.2
% 1
house = imread('house.tif');
house = double(house);
% 2
ls_version = 255*(house/255).^2.2;
% 3
I2 = [1,2;3,0];
I4 = [4*I2+1,4*I2+2;4*I2+3,4*I2];
I8 = [4*I4+1,4*I4+2;4*I4+3,4*I4];
% 4
T1 = 255 * (I2+0.5) / (2*2);
T2 = 255 * (I4+0.5) / (4*4);
T3 = 255 * (I8+0.5) / (8*8);

[rowh,colh] = size(house);
b1 = zeros(rowh,colh);
b2 = zeros(rowh,colh);
b3 = zeros(rowh,colh);
```

```
% 2x2
for i = 1:rowh
    for j = 1:colh
        if ls_version(i,j) > T1(mod(i-1,2)+1,mod(j-1,2)+1)
            b1(i,j) = 255;
        end
    end
end
% 4x4
for i = 1:rowh
    for j = 1:colh
        if ls_version(i,j) > T2(mod(i-1,4)+1,mod(j-1,4)+1)
            b2(i,j) = 255;
        end
    end
end
% 8x8
for i = 1:rowh
    for j = 1:colh
        if ls_version(i,j) > T3(mod(i-1,8)+1,mod(j-1,8)+1)
            b3(i,j) = 255;
        end
    end
end
% 5
figure()
imwrite(b1,'421.tiff')
imshow(b1)
truesize

figure()
imwrite(b2,'422.tiff')
imshow(b2)
truesize

figure()
imwrite(b3,'423.tiff')
imshow(b3)
truesize
% 6
square1 = (b1-house).^2;
RMSE1 = sqrt(1/(rowh*colh) * sum(square1(:)));
fid1 = fidelity(house,b1);

square2 = (b2-house).^2;
RMSE2 = sqrt(1/(rowh*colh) * sum(square2(:)));
fid2 = fidelity(house,b2);
```

```
square3 = (b3-house).^2;
RMSE3 = sqrt(1/(rowh*colh) * sum(square3(:)));
fid3 = fidelity(house,b3);
%}

%% 5.1
% 1
house = imread('house.tif');
house = double(house);
ls_version = 255*(house/255).^2.2;
[rowh,colh] = size(house);
output = zeros(rowh,colh);
% 2,3,4,5
T = 127; % threshold
for i = 1:rowh-1
    for j = 2:colh-1
        if ls_version(i,j) > T
            output(i,j) = 255;
            error = ls_version(i,j) - output(i,j);
        else
            output(i,j) = 0;
            error = ls_version(i,j) - output(i,j);
        end
        ls_version(i,j+1) = 7/16*error + ls_version(i,j+1);
        ls_version(i+1,j-1) = 3/16*error + ls_version(i+1,j-1);
        ls_version(i+1,j) = 5/16*error + ls_version(i+1,j);
        ls_version(i+1,j+1) = 1/16*error + ls_version(i+1,j+1);
    end
end
figure()
imshow(output)
truesize
imwrite(output,'5.tiff')

square = (output-house).^2;
RMSE = sqrt(1/(rowh*colh) * sum(square(:)));
fid = fidelity(house,output);
```

```
function fid=fidelity(f,b)
% a
f = 255*(f/255).^2.2;
b = 255*(b/255).^2.2;

h = zeros(7);
theta_sq = 2;
for i = -3:3
    for j = -3:3
        h(i+4,j+4) = exp(-(i^2+j^2)/(2*theta_sq));
    end
end
h = h/sum(h(:));
f = imfilter(f,h);
b = imfilter(b,h);
% c
f = 255*(f/255).^(1/3);
b = 255*(b/255).^(1/3);

[rowf,colf] = size(f);
square = (f-b).^2;
fid = sqrt(1/(rowf*colf) * sum(square(:)));
end
```