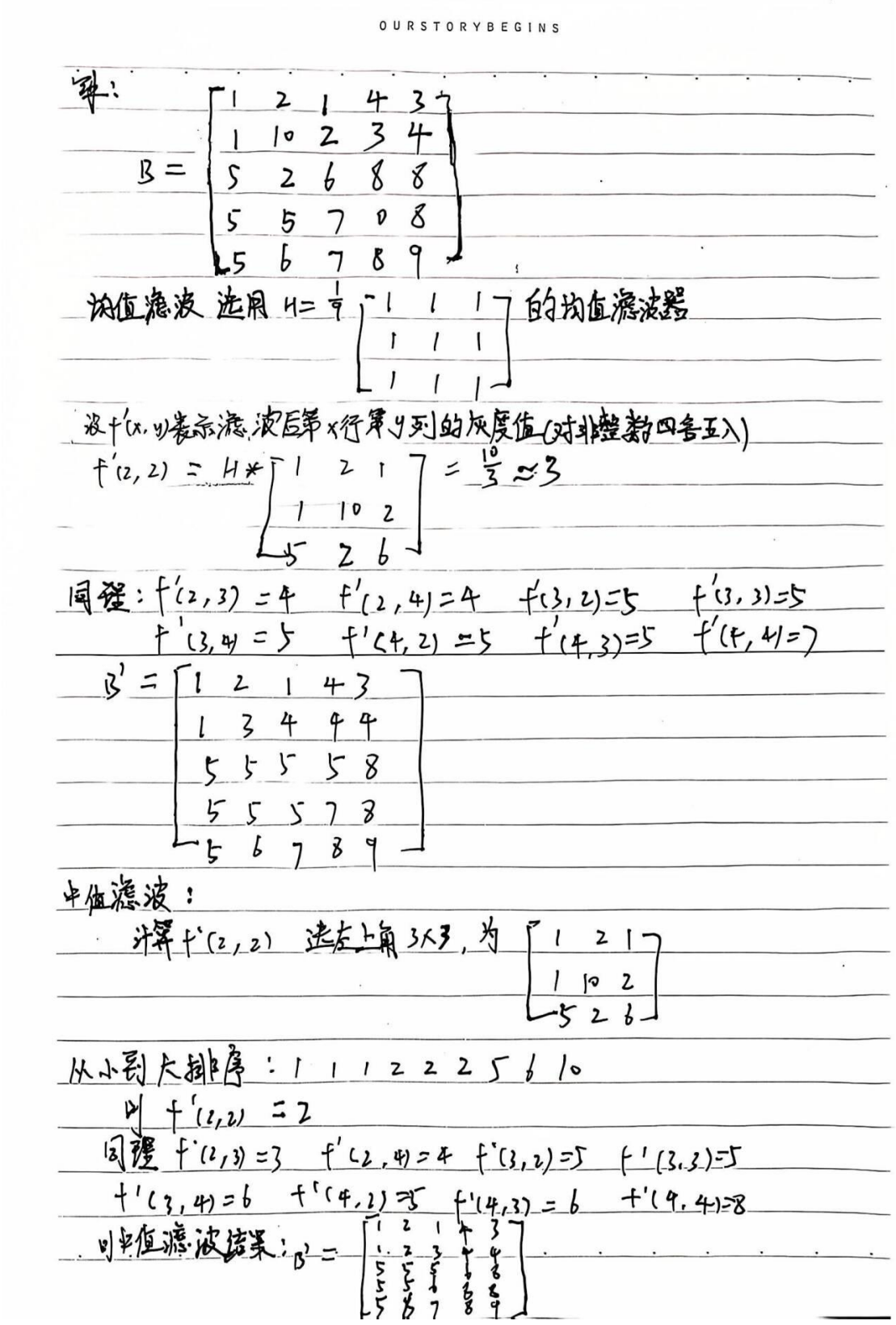
电子签名：

一、理论作业


OURSTORYBEGINS
解：
$B=\begin{bmatrix}1&2&1&4&3\\1&10&2&3&4\\5&2&6&8&8\\5&5&7&0&8\\5&6&7&8&9\end{bmatrix}$
均值滤波 选用 $H=\frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$ 的均值滤波器
设 $f'(x,y)$ 表示滤波后第 x 行第 y 列的灰度值（对非整数四舍五入）
$f'(2,2)=H*\begin{bmatrix}1&2&1\\1&10&2\\5&2&6\end{bmatrix}=\frac{10}{3}\approx 3$
同理：$f'(2,3)=4$ $f'(2,4)=4$ $f'(3,2)=5$ $f'(3,3)=5$
$f'(3,4)=5$ $f'(4,2)=5$ $f'(4,3)=5$ $f'(4,4)=7$
$B'=\begin{bmatrix}1&2&1&4&3\\1&3&4&4&4\\5&5&5&5&8\\5&5&5&7&8\\5&6&7&8&9\end{bmatrix}$
中值滤波：
计算 $f'(2,2)$ 选左上角 3×3，为 $\begin{bmatrix}1&2&1\\1&10&2\\5&2&6\end{bmatrix}$
从小到大排序：1 1 1 2 2 2 5 6 10
则 $f'(2,2)=2$
同理 $f'(2,3)=3$ $f'(2,4)=4$ $f'(3,2)=5$ $f'(3,3)=5$
$f'(3,4)=6$ $f'(4,2)=5$ $f'(4,3)=6$ $f'(4,4)=8$
则中值滤波结果：$B'=\begin{bmatrix}1&2&1&4&3\\1&2&3&4&4\\5&5&5&6&8\\5&6&7&8&9\end{bmatrix}$

二、编程作业

2.1 灰度直方图与离散傅里叶变换幅度图

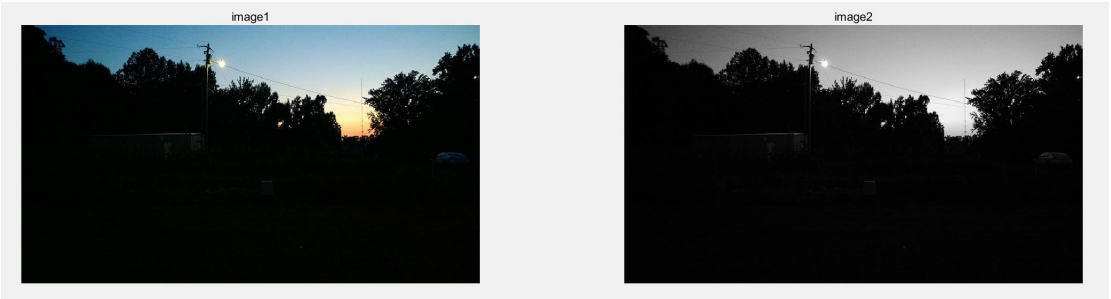
2.1.1 代码实现

```
% 转换为灰度图
%编程转化为灰度图
img=imread('image1.jpg');
gray_img = rgb2gray(img);
figure,
subplot(121);imshow(img);title('image1');
subplot(122);imshow(gray_img);title('image2');    % 显示图像并设置标题
%输入图像
[m,n]=size(gray_img); % 计算图像的长宽
p=zeros(1,256); %创建数组存储像素个数
%显示直方图
for i=0:255
p(i+1)=length(find(img==i));
end
figure;bar([0:255],p);
% 计算二维离散傅里叶变换（DFT）
perform_dft(gray_img);
% 函数实现
function magnitude_spectrum = perform_dft(gray_img)
% 下面是傅里叶正变换必备的一些矩阵:
[M,N]=size(gray_img);
Wm = exp(-1i*2*pi/M);
Wn = exp(-1i*2*pi/N); % 不同 G 中用不同的 W
Em = zeros(M);
En = zeros(N); % E 是辅助计算矩阵
Gm = zeros(M)+Wm;
Gn = zeros(N)+Wn; % G 是计算时要用的矩阵
F = zeros(M,N); % F 是转换到频域的结果
E = zeros(M,N);
% 对 Gm 的计算: 循环长度为 M
fprintf('二维离散傅里叶变换开始:\n');
```

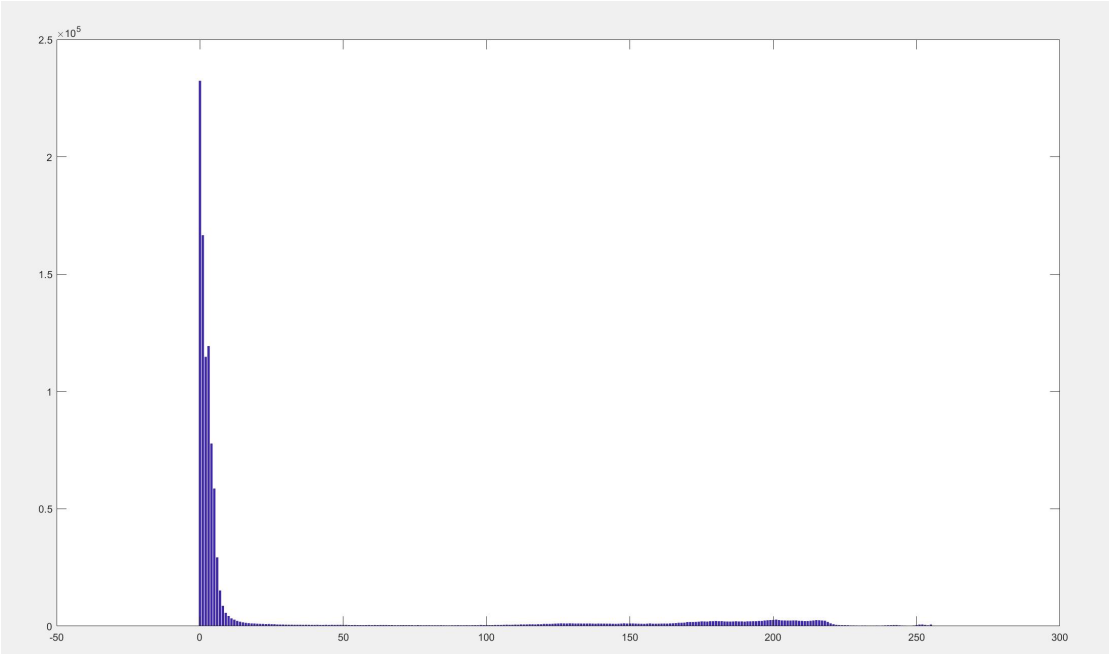
```
for row = 0:M-1
for col = 0:M-1
Em(row+1,col+1) = row * col;
Gm(row+1,col+1) = Gm(row+1,col+1)^Em(row+1,col+1);
end
end
%变换到图像中点
% 对 Gn 的计算: 循环长度为 N
for row = 0:N-1
for col = 0:N-1
En(row+1,col+1) = row * col;
Gn(row+1,col+1) = Gn(row+1,col+1)^En(row+1,col+1);
end
end
for row =1:M
for col =1:N
E(row,col)=double(gray_img(row,col))*((-1)^(row+col));
end
end
F = real(Gm*E*Gn);
% 计算幅度谱并进行对数变换以增强可视化效果
magnitude_spectrum = log(1 + abs(F));
subplot(1, 2, 1), imshow(gray_img), title('Original Grayscale Image');
subplot(1, 2, 2), imshow(magnitude_spectrum, []), title('Magnitude
Spectrum');
end
```

2.1.2 实验结果

原图与灰度图：



灰度直方图：



灰度图与傅里叶变换幅度图:

2.2 直方图均衡化和同态滤波

2.2.1 代码实现

```
% 转换为灰度图
%编程转化为灰度图
img=imread('image1.jpg');
gray_img = rgb2gray(img);
%直方图均衡化
hestepimg(gray_img)
%函数实现
function his=hestepimg(img)
```

```
% 输入图像
[m,n]=size(img); % 计算图像的长宽
p=zeros(1,256); %创建数组存储像素概率
% 统计每个像素值出现的概率， 得到概率直方图
% 用 length 函数计算相同像素的个数
for i=0:255
p(i+1)=length(find(img==i))/(m*n);
end
% 求累计概率，得到累计直方图
s=zeros(1,256);
for i=1:256
for j=1:i
s(i)=p(j)+s(i);
end
end
%完成每个像素点的映射
a=round(s*255); %四舍五入
his=img;
for i=0:255
his(img==i)=a(i+1);
end
%均衡化后的直方图
k=zeros(1,256); %创建数组存储像素概率
% 用 length 函数计算相同像素的个数
for i=0:255
k(i+1)=length(find(his==i));
end
%展示均衡化前后图片
figure;
subplot(121),imshow(img);title('原图')
subplot(122),imshow(his);title('均衡化后')
figure;bar([0:255],k);%展示直方图均衡化后的灰度直方图
end
%同态滤波
homomorphic(gray_img)
%函数实现
function imgtemp=homomorphic(image)
%参数声明
rH = 0.15;
rL = 0.1;
c = 0.15;%介于 rH 和 rL 之间
D0 = 0.2;
[M, N] = size(image);
%取对数
```

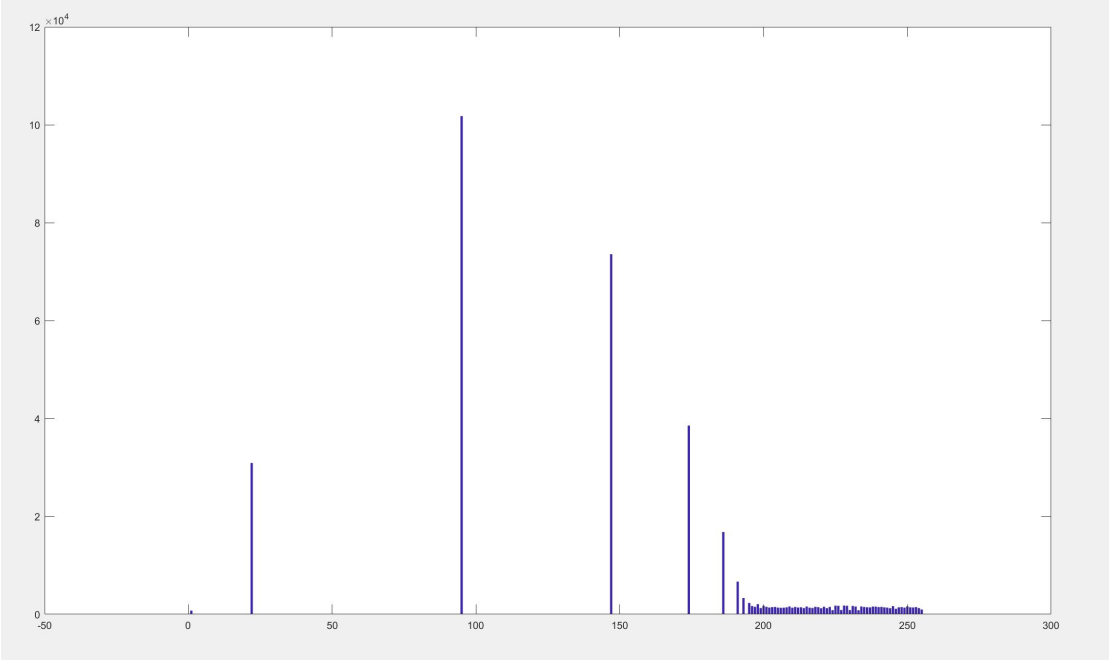
```
img_log = log(double(image) + 1);
%平移到中心，判断语句代替指数计算
img_py = zeros(M, N);
for i = 1:M
for j= 1:N
if mod(i+j, 2) == 0
img_py(i,j) = img_log(i, j);
else
img_py(i,j) = -1 * img_log(i, j);
end
end
end
% 对平移后的图像进行傅里叶变换
img_py_fft = fft2(img_py);
%同态滤波函数
%滤波器选用高斯滤波
H = zeros(M, N);
r = rH - rL;
D = D0^2;%设置参数
m_mid=floor(M/2);%中心点坐标
n_mid=floor(N/2);
for i = 1:M
for j =1:N
dis = ((i-m_mid)^2+(j-n_mid)^2);%到中心店的距离
H(i, j) = r * (1-exp((-c)*(dis/D))) + rL;%高斯同态滤波函数
end
end
imgtemp = img_py_fft.*H;%滤波
%开始反变换
imgtemp = abs(real(ifft2(imgtemp)));
imgtemp = exp(imgtemp) - 1;
%归一化处理
max_num = max(imgtemp(:));min_num = min(imgtemp(:));
range = max_num - min_num;
img_after = zeros(M,N,'uint8');
for i = 1 : M
for j = 1 : N
img_after(i,j) = uint8(255 * (imgtemp(i, j)-min_num) / range);
end
end
%同态滤波后的直方图
k=zeros(1,256); %创建数组存储像素概率
% 用 length 函数计算相同像素的个数
for i=0:255
```

```
k(i+1)=length(find(img_after==i));
end
figure;%展示同态滤波前后照片
subplot(1,2,1), imshow(image), title('原图像');
subplot(1,2,2), imshow(img_after), title('变换后');
figure;bar([0:255],k);
end
```

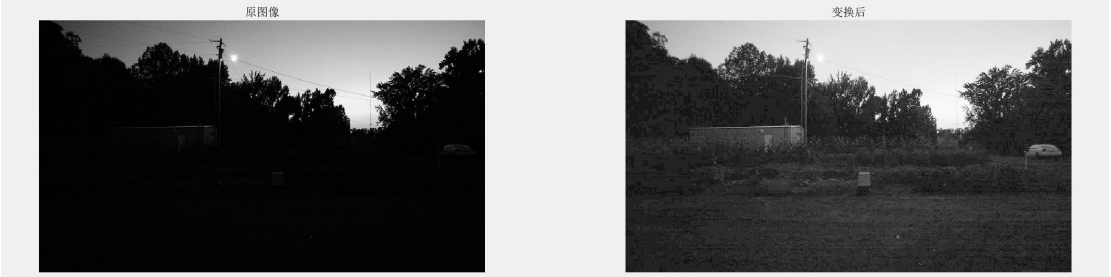
2.2.2 实验结果

灰度图与均衡化后图像：

均衡化后灰度直方图：



灰度图与同态滤波图：



同态滤波后灰度直方图：