Project3
Filtering in Frequency Domain

张元鑫 2018210902

实验原理

（1）Homomorphic Filtering

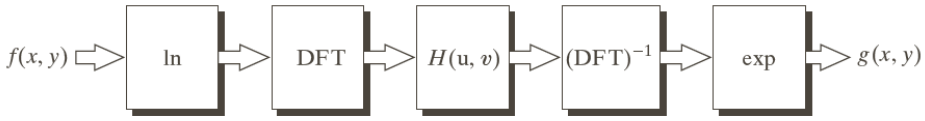


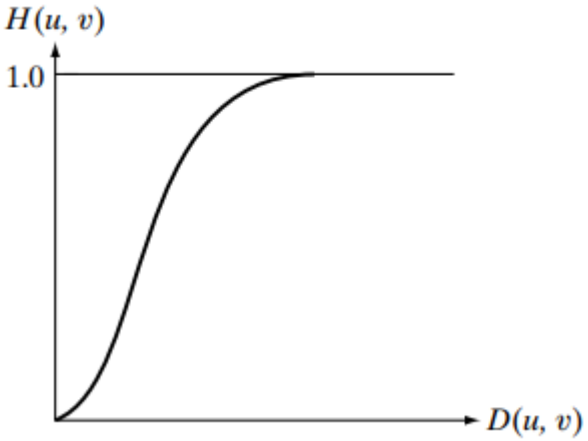
FIGURE 4.60 Summary of steps in homomorphic filtering.

先对图像函数 f(x,y)取对数，然后做二维 DFT 变换的到图像的频域函数 F(x,y)，之后对频域函数 F 乘以频域高通滤波器 H(u,v)，然后做反 DFT 变换,最后通过 exp 函数恢复原图像。

实验中使用的同态高通滤波器，其表达式为：

$$\mathrm{H(u,v)} = (\gamma_{\mathrm{H}} - \gamma_{\mathrm{L}})\left[1 - \mathrm{e}^{-\mathrm{c}\left[\frac{\mathrm{D}^2(\mathrm{u,v})}{\mathrm{D}_0^2}\right]}\right] + \gamma_L$$

其中$\mathrm{D(u,v)} = \sqrt{(\mathrm{x-u})^2 + (y-v)^2}$ 是距离函数



关键代码：

```
rL=2.8; %
rH=4.5;
c =1;
D0=80;
xc=floor(m/2);
yc=floor(n/2);
D = zeros([m,n]);
H = zeros([m,n]);
for i=1:m
    for j=1:n
        dmin = min([i^2+j^2,i^2+(j-n)^2,(i-m)^2+j^2,(i-m)^2+(j-n)^2]);
        D(i,j)=sqrt(dmin);   %compute the required distances
        H(i,j)=(rH-rL).*(1-exp(-
c*(D(i,j)^2./D0^2)))+rL; %the Gausian high-pass filter
    end
end
```

（2）DFT 和 IDFT 的实现

可以直接根据 DFT 的定义来实现 DFT 函数

$$\mathrm{F(u,v)} = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$$

对于图像的二维 DFT 变换，可以先对原图像的每一行做 1 维 DFT 变换，然后对变换后的矩阵的每一列再做 1 次 1 维 DFT。

关键代码：

```
WN_r=exp(-1i*2*pi/N);
WN_c=exp(-1i*2*pi/M);
nk_r=n'*k;
mp_c=m'*p;
WNnk_r=WN_r.^nk_r;
WNmp_c=WN_c.^mp_c;
F = zeros([M,N,dim]);
for i=1:M
    for k=1:dim
        F(i,:,k) = I(i,:,k)*WNnk_r;
    end
end
for j=1:N
    for k=1:dim
        F(:,j,k) = WNmp_c*F(:,j,k);
    end
end
```

IDFT 的公式为：

$$f(\mathrm{x},\mathrm{y}) = \frac{1}{\mathrm{MN}} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} f(u,v) e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$$

关键代码：

```
WN_r=exp(-1i*2*pi/N);
WN_c=exp(-1i*2*pi/M);
nk_r=n'*k;
mp_c=m'*p;
WNnk_r=WN_r.^-nk_r;
WNmp_c=WN_c.^-mp_c;
F = zeros([M,N,dim]);
for i=1:M
    for k=1:dim
        F(i,:,k) = I(i,:,k)*WNnk_r/N;
    end
end
for j=1:N
    for k=1:dim
        F(:,j,k) = WNmp_c*F(:,j,k)/M;
    end
end
```

（3）直方图均衡

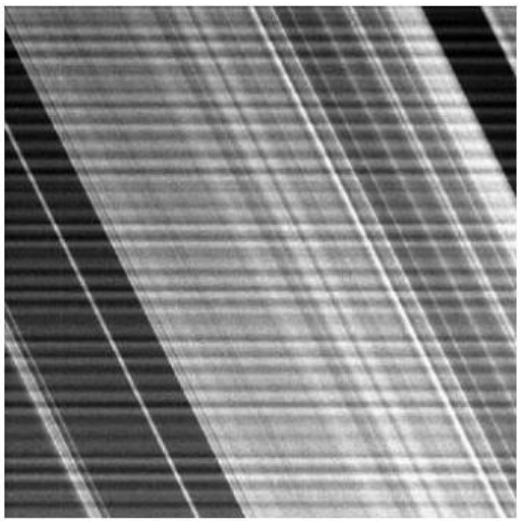
先求原始图像的直方图：

```
H = zeros([1,256]); %the gray histogram vector
for i=1:m
    for j=1:n
        H(A(i,j)+1)=H(A(i,j)+1)+1;%the coresponding va
lue plus 1
    end
end
H = H/(m*n);%normalize the histogram
```

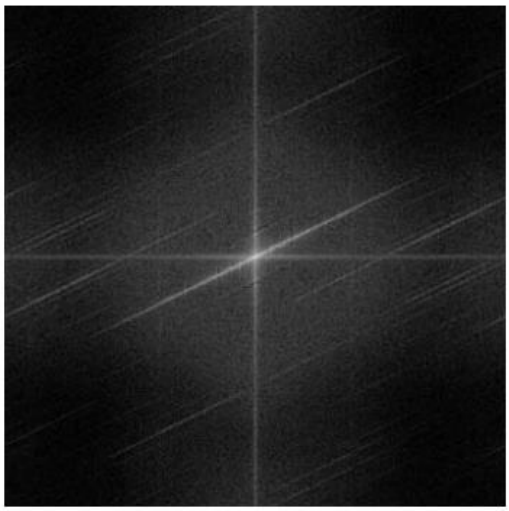
之后求原始图像灰度级概率密度的累积，也就是新图像灰度级的概率密度，再乘以 255 就得到新图象的灰度值：

```
S=zeros(1,256); %Sk is the cumulative histogram
for k=1:256
    S(k)=sum(H(1:k));
end
S = 255*S; %S is the new gray histogram of I
```

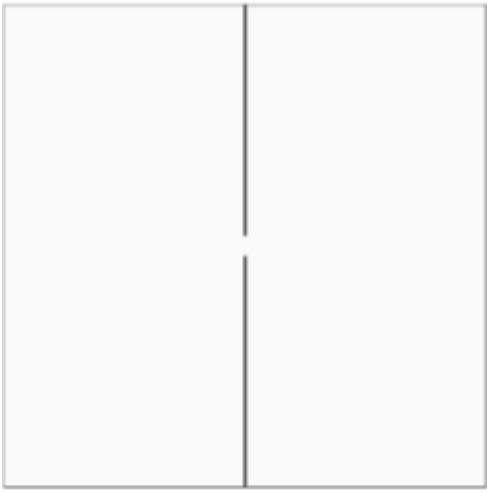
（4）Notch filter



原图中存在由于垂直的正弦噪声，将图片进行 DFT 变换到频域，观察到其频域图像为，中间窄窄的竖直区域就是这个正弦噪声的频域分布。



因此可以设计一个如下图所示的陷波滤波器，即可滤除这个噪声。



关键代码：

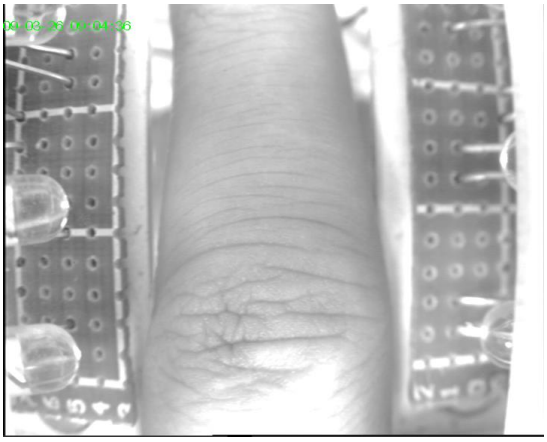
```
H = zeros([m,n])+1;
H(:,1)=0;
H(:,n)=0;
H(1:10,:)=1;
H(m-9:m,:)=1;
IF = H.*I_f;
```

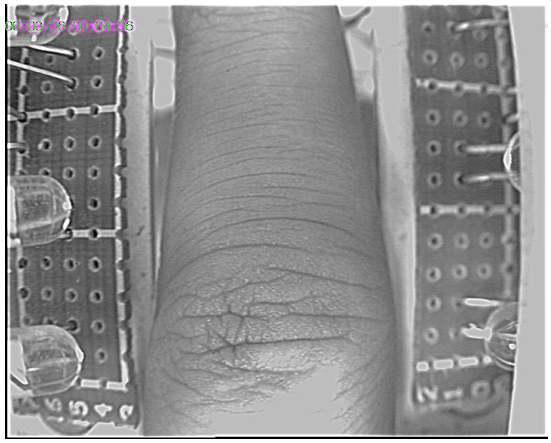
实验结果

Task(1):

第一张图是一群人的合影，左下是原图，可见原图有一部分光照反射极强，一部分则非常暗。通过同态滤波器后的效果如右图所示。

第二张图是一张整体亮度都非常大的图片，通过同态滤波器后的结果如右下图所示。

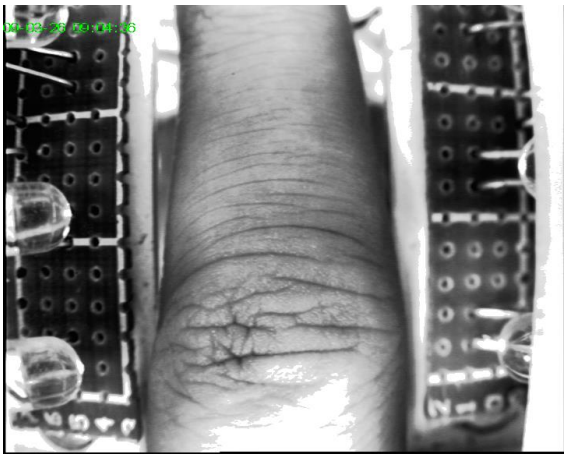




完整代码见 task1.m

Task(2):

使用直方图均衡法处理得到的两张图如下所示，与同态滤波器相比，直方图均衡法不需要手动调整参数，表现效果也相对更好一些。



完整代码见 task2.m 和 myhisteq.m

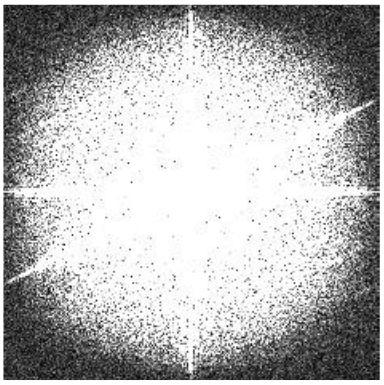
Task(3):

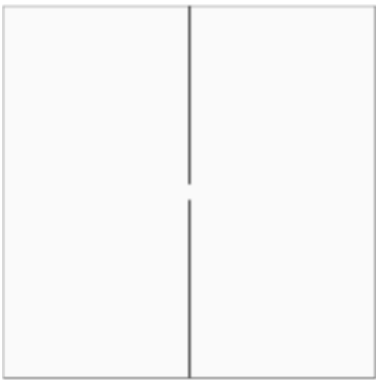
实现了自己编写的 DFT2 和 IDFT2 功能，与 matlab 系统自带的 fft2 和 ifft2 函数对比结果相同。

完整代码见 myDFT2.m 和 myIDFT2.m

Task(4):

下左图是原图做了 DFT 变换后得到的频域图像，下右图是使用的陷波滤波器频域图像。





下图是经过滤波器处理后的图像，可见正弦噪声已被很好地消除。