**Project3**

**Filtering in Frequency Domain**

张元鑫 2018210902

**实验原理**

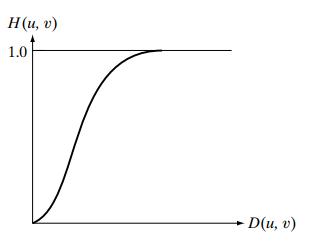
1. Homomorphic Filtering



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：

其中是距离函数



关键代码：

1. rL=2.8; %
2. rH=4.5;
3. c =1;
4. D0=80;
5. xc=floor(m/2);
6. yc=floor(n/2);
7. D = zeros([m,n]);
8. H = zeros([m,n]);
9. **for** i=1:m
10. **for** j=1:n
11. dmin = min([i^2+j^2,i^2+(j-n)^2,(i-m)^2+j^2,(i-m)^2+(j-n)^2]);
12. D(i,j)=sqrt(dmin);   %compute the required distances
13. H(i,j)=(rH-rL).\*(1-exp(-c\*(D(i,j)^2./D0^2)))+rL; %the Gausian high-**pass** filter
14. end
15. end
16. DFT和IDFT的实现

可以直接根据DFT的定义来实现DFT函数

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：

1. WN\_r=exp(-1i\*2\*pi/N);
2. WN\_c=exp(-1i\*2\*pi/M);
3. nk\_r=n'\*k;
4. mp\_c=m'\*p;
5. WNnk\_r=WN\_r.^nk\_r;
6. WNmp\_c=WN\_c.^mp\_c;
7. F = zeros([M,N,dim]);
8. **for** i=1:M
9. **for** k=1:dim
10. F(i,:,k) = I(i,:,k)\*WNnk\_r;
11. end
12. end
13. **for** j=1:N
14. **for** k=1:dim
15. F(:,j,k) = WNmp\_c\*F(:,j,k);
16. end
17. end

IDFT的公式为：

关键代码：

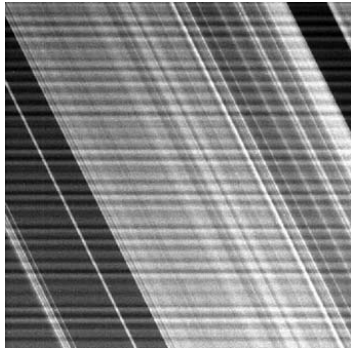
1. WN\_r=exp(-1i\*2\*pi/N);
2. WN\_c=exp(-1i\*2\*pi/M);
3. nk\_r=n'\*k;
4. mp\_c=m'\*p;
5. WNnk\_r=WN\_r.^-nk\_r;
6. WNmp\_c=WN\_c.^-mp\_c;
7. F = zeros([M,N,dim]);
8. **for** i=1:M
9. **for** k=1:dim
10. F(i,:,k) = I(i,:,k)\*WNnk\_r/N;
11. end
12. end
13. **for** j=1:N
14. **for** k=1:dim
15. F(:,j,k) = WNmp\_c\*F(:,j,k)/M;
16. end
17. end
18. 直方图均衡

先求原始图像的直方图：

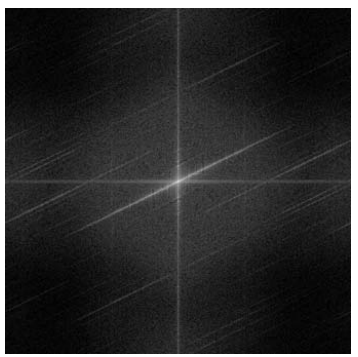
1. H = zeros([1,256]); %the gray histogram vector
2. **for** i=1:m
3. **for** j=1:n
4. H(A(i,j)+1)=H(A(i,j)+1)+1;%the coresponding value plus 1
5. end
6. end
7. H = H/(m\*n);%normalize the histogram

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：

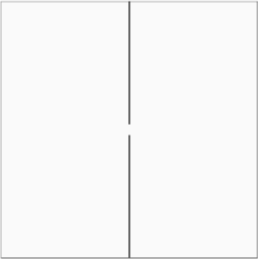
1. S=zeros(1,256); %Sk is the cumulative histogram
2. **for** k=1:256
3. S(k)=sum(H(1:k));
4. end
5. S = 255\*S; %S is the **new** gray histogram of I
6. Notch filter



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：

1. H = zeros([m,n])+1;
2. H(:,1)=0;
3. H(:,n)=0;
4. H(1:10,:)=1;
5. H(m-9:m,:)=1;
6. IF = H.\*I\_f;

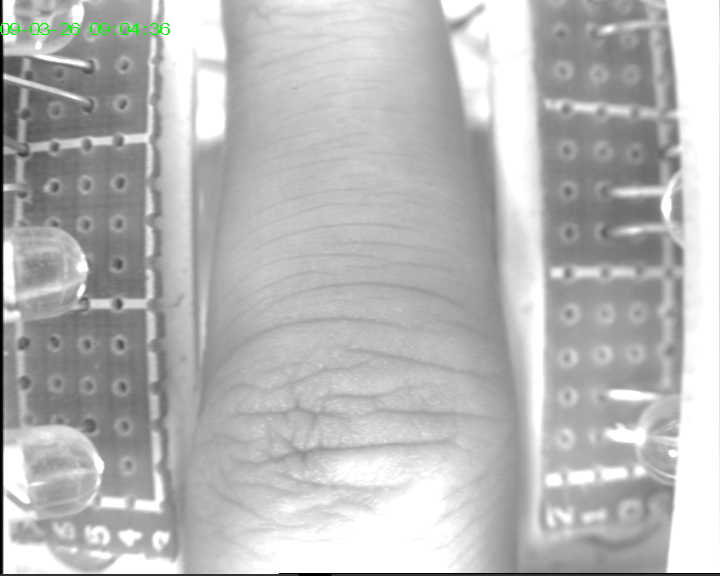
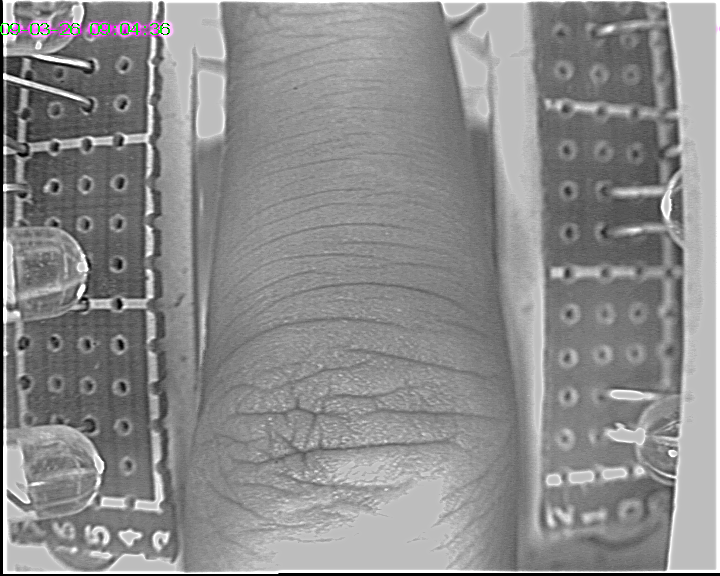
**实验结果**

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。

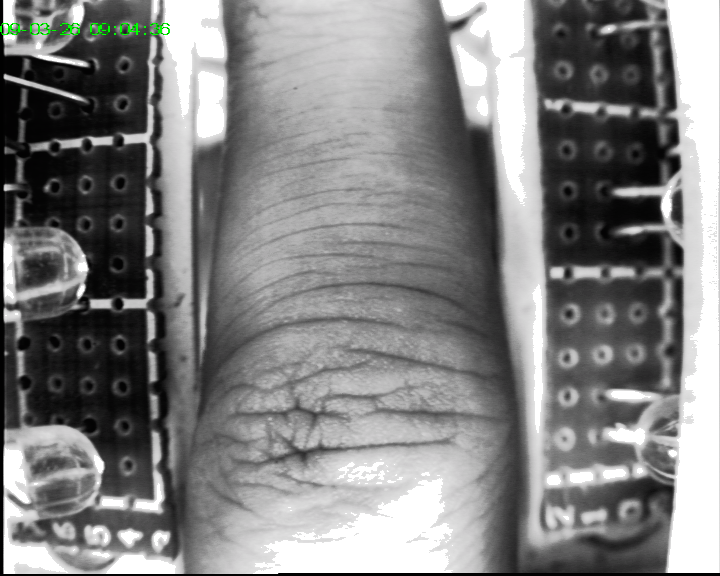
 

**完整代码见task1.m**

**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

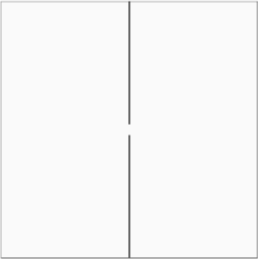
**Task(3):**

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。

