

Matlab:同态滤波



📅 2020-10-06 | 📁 [Matlab图像处理](#) |

一、什么是同态滤波

根据成像原理，一幅图像可以分成两种成分的乘积叠加，一种成分是入射的光强 $L(x,y)$ ，另一种是反射到人眼中的光强 $R(x,y)$ ，二者是乘性叠加，所以一幅图像可以表示为 $F(x,y)=L(x,y)*R(x,y)$ 。

同态滤波通过将图像取对数， $\ln(F(x,y))=\ln(L(x,y))+\ln(R(x,y))$ ，将乘性耦合的入射光和反射光成分解耦，变成加性耦合，这样就可以利用线性性分别对两种成分进行滤波处理。由于反射光强一般是高频成分，而入射光强是低频成分，所以用高通滤波器可以将入射光成分滤掉，得到更加清楚的细节。同态滤波是通过物理规律将图像进行分解操作，滤波器是在滤波过程中的一种频率选择方法，同态滤波并没有设计一种新的滤波器。

二、同态滤波实现

同态滤波结构

```
1 homomorphic filter
2 |
3 |-log_img = log(img + 1)
4 |
5 |-ft_img = fft(img)
6 |
7 |-h_img = ft_img.*H
8 |
9 |-ift_img = ifft(h_img)
10 |
11 |-res_img = exp(ift_img) - 1
```

matlab代码

```
1 function [ output_img ] = homomorphic_filt( input_img, rL, rH, c, d )
2
3     %同态滤波参数设置
4     %rL = 0.4; %低频放大系数
5     %rH = 10; %高频放大系数
6     %c = 0.2; %高斯变换常数系数
7     %d = 2000; %高斯变换系数
8
9     [M,N] = size(input_img);
10    input_img = double(input_img);
11
12    log_img = log(input_img+1);
13    ft_img = fft2(log_img);
14
15    %构造高斯滤波器
16    H = zeros(M,N);
17    for i = 1:M
18        for j = 1:N
19            D = (i.^2+j.^2);
20            H(i,j) = (rH-rL).*(1-exp(-c.*(D./(d^2))))+rL;
```

```
21         end
22     end
23
24     filtered_img = H.*ft_img;
25     ift_img = ifft2(filtered_img);
26     exp_img = exp(ift_img)-1;
27
28     output_img = uint8(abs(exp_img));
29
30 end
```

有关中心化的处理

参考了许多CSDN网站上的博客，发现他们都有将高斯滤波器中心化的操作，但是我认为这是不必要的操作，仅仅是坐标轴的变换而已。

我们来分类讨论一下，如果图像的傅里叶变换没有进行平移操作（fftshift），那么结果的左上角是低频成分，沿着x、y轴坐标增加的方向，频率逐渐增高，那么高通滤波器应该如何设置呢？以左上角为原点，到远点的距离为变量即可构造高斯高通滤波器。但是如果以图片中心点为原点，这时候H需要进行平移，将它和图像的傅里叶变换结果的高低频区域对齐。

三、实验结果



原图

gray img



灰度图

homo filtered img



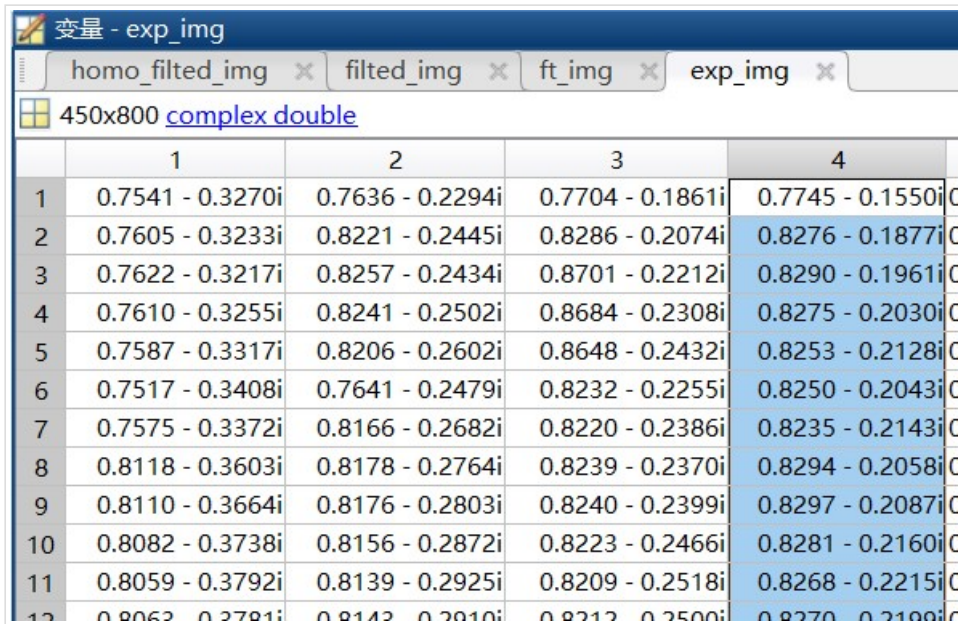
变换后的图

可以看出变换后的图像有明显的二值化现象，图片的展现效果不好。

四、失误与总结

图像二值化的原因及改进

在调试过程中，发现取指数后的矩阵元素的值非常小，所以在之后的取整等操作无法反映图像的真实灰度，因此需要将取对数后的图像的灰度拓展到0到255。改进策略是将指数图像的最大最小值取出，然后将指数图像与最小值的差按比例放大到0到255。



	1	2	3	4
1	0.7541 - 0.3270i	0.7636 - 0.2294i	0.7704 - 0.1861i	0.7745 - 0.1550i
2	0.7605 - 0.3233i	0.8221 - 0.2445i	0.8286 - 0.2074i	0.8276 - 0.1877i
3	0.7622 - 0.3217i	0.8257 - 0.2434i	0.8701 - 0.2212i	0.8290 - 0.1961i
4	0.7610 - 0.3255i	0.8241 - 0.2502i	0.8684 - 0.2308i	0.8275 - 0.2030i
5	0.7587 - 0.3317i	0.8206 - 0.2602i	0.8648 - 0.2432i	0.8253 - 0.2128i
6	0.7517 - 0.3408i	0.7641 - 0.2479i	0.8232 - 0.2255i	0.8250 - 0.2043i
7	0.7575 - 0.3372i	0.8166 - 0.2682i	0.8220 - 0.2386i	0.8235 - 0.2143i
8	0.8118 - 0.3603i	0.8178 - 0.2764i	0.8239 - 0.2370i	0.8294 - 0.2058i
9	0.8110 - 0.3664i	0.8176 - 0.2803i	0.8240 - 0.2399i	0.8297 - 0.2087i
10	0.8082 - 0.3738i	0.8156 - 0.2872i	0.8223 - 0.2466i	0.8281 - 0.2160i
11	0.8059 - 0.3792i	0.8139 - 0.2925i	0.8209 - 0.2518i	0.8268 - 0.2215i

取指数后的图像矩阵部分元素

- 改进后的代码

```

1 function [ output_img ] = homomorphic_filt( input_img, rL, rH, c, d )
2
3 %同态滤波参数设置
4 %rL = 0.1; %低频放大系数
5 %rH = 5; %高频放大系数
6 %c = 0.2; %高斯变换常数系数
7 %d = 1000; %高斯变换系数
8
9 [M,N] = size(input_img);
10 input_img = double(input_img);
11
12 log_img = log(input_img+1);
13 ft_img = fft2(log_img);
14
15 %构造高斯滤波器
16 H = zeros(M,N);
17 for i = 1:M
18     for j = 1:N
19         D = (i.^2+j.^2);
20         H(i,j) = (rH-rL).*(1-exp(-c.*(D./(d^2))))+rL;
21     end
22 end
23
24 filted_img = H.*ft_img;
25 ift_img = ifft2(filted_img);
26 exp_img = exp(ift_img);
27 output_img =
28     uint8( 255 * ( exp_img - ones(M,N)*min(min(exp_img)) )
29         / ( max(max(exp_img)) - min(min(exp_img)) ) );
30 %output_img = uint8(abs(exp_img));
31 end

```

- 改进后实验结果

homo filtered img



改进后变换后的图

调参

这次的实验参数较多，调参的过程比设计算法的过程还要长，但是通过调参也能够理解各个参数的对图像的影响。比如d和rH的作用又互相抵消的效果，c参数会让图片整体明暗发生变化。至于为什么会出现这些现象，还是要从原理入手，这里还需研究。后续如果有可能的话，可以制作四个参数在一定范围内变化的图像效果变化动图，更加直观的显示四个参数的影响。

洪嘉豪

Matlab

图像处理

© 2020  Darcy

Powered by [Hexo](#) | Theme — [NexT.Mist](#) v5.1.4

