数字图像处理 project

Project title:	Histogram Equalization	Project number:	Project 0403
姓　　名:	彭思达	学　　号:	3140103545
学　　院:	信电学院	专　　业:	信息工程
Date due:	2017.5.1	Date handed in:	April 10, 2017

Abstract

这篇报告实现了二维高斯低通滤波器。首先文章讨论了一维频率域高斯滤波器于空间域中相应滤波器的对应关系，得出结论：高斯低通滤波器越窄，其衰减的低频越多，引起的模糊越大。在空间域，这意味着必须使用较大的模板来增加模糊。

随后文中实现了二维高斯低通滤波器，形式为 $\boldsymbol{H(u,v) = e^{-\frac{D^2(u,v)}{2D_0^2}}}$ 。其中滤波器的大小为 $\boldsymbol{M \times N}$，M 为图像的两倍宽，N 为图像的两倍长。高斯低通滤波器的中心由距离公式 $\boldsymbol{D^2(u,v)}$ 确定，$\boldsymbol{D^2(u,v) = (u - M/2)^2 + (v - N/2)^2}$。

为了感受二维高斯低通滤波器的效果，本文用代码实现了变换函数，并以书上 fig 4.41(a) 为实验对象，得到它原图像的频谱、相应输出图像的频谱及输出图像。在文章的最后附上了相应的实现代码。

April 10, 2017

1 Technical discussion

1.1 一维高斯低通滤波器与空间滤波器的对应关系

令 $H(u)$ 表示一维频率域高斯滤波器:

$$H(u) = Ae^{-u^2/2\sigma^2} \tag{1}$$

空间域中相应的滤波器可由 $H(u)$ 的傅立叶反变换得到:

$$h(u) = \sqrt[2]{2\pi}\sigma Ae^{-2\pi^2\sigma^2x^2} \tag{2}$$

观察空间域中的滤波器可以得出两点结论:

1. 频率域高斯低通滤波器和空间域高斯低通滤波器的值全是正的，所以我们应该使用一个全部带正系数的模板在空间域中实现低通滤波。
2. 频率域滤波器越窄，空间域中滤波器就越宽，这意味着必须使用更大的模板来增加模糊。

1.2 二维高斯低通滤波器的讨论

高斯低通滤波器的二维形式由下式给出:

$$H(u,v) = e^{-\frac{D^2(u,v)}{2D_0^2}} \tag{3}$$

其中，D_0 是截止频率，$D(u,v) = (u - M/2)^2 + (v - N/2)^2$ 是距离公式。滤波器的大小 $M \times N$ 由输入图像的大小确定，M 为图像的两倍宽，N 为图像的两倍长。

因为 GLFT 的傅立叶反变换也是高斯的，所以通过 IDFT 得到的空间高斯滤波器没有振铃现象。

2 Discussion of results

一维高斯低通滤波器与空间滤波器的对应关系如下:

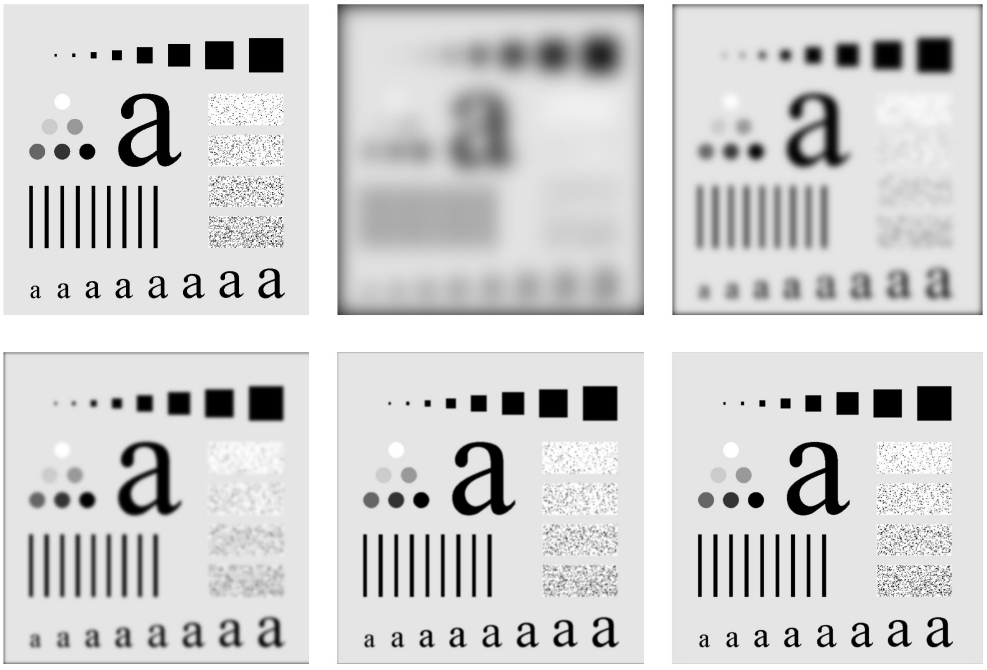
1. 频率域高斯低通滤波器和空间域高斯低通滤波器的值全是正的，所以我们应该使用一个全部带正系数的模板在空间域中实现低通滤波。
2. 频率域滤波器越窄，空间域中滤波器就越宽，这意味着必须使用更大的模板来增加模糊。

文中使用的二维高斯低通滤波器有如下特性：

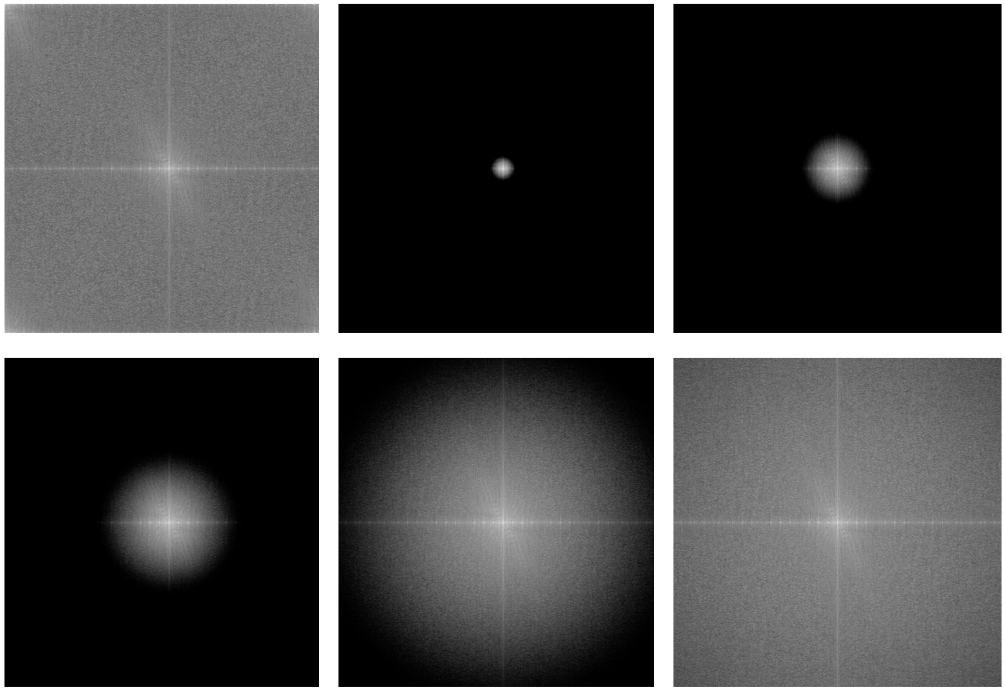
1. 滤波器的大小 $M \times N$ 由输入图像的大小确定，M 为图像的两倍宽，N 为图像的两倍长。
2. 滤波器的中心为 $(M/2, N/2)$。
3. 频率域中的高斯低通滤波器通过 IDFT 得到的空间高斯滤波器没有振铃现象。

3 Results

下面列出原图和处理后的六张图，第一张是原图，然后从左到右，从上到下，每张图的对应滤波器的截止频率分别设置在半径值 10，30，60，160，460：



原图的频谱与处理后的频谱如下所示：



4 Appendix

实现二维高斯低通滤波器的代码如下：

```
function [img] = low_filter( img, D0 )
%LOW_FILTER Summary of this function goes here
%   Detailed explanation goes here

% 用零填充图像
img = fill(img);
% 将图像移到(P/2, Q/2)这个中心
img = shift(img);
% 将图像进行傅立叶变换
img = fft2(img);
% 显示图像的频谱
imshow(log(1+abs(img)), []);

% 获得高斯低通滤波器
H = GLPF(img, D0);
% H = GHPF(img, D0);

% 在频率域用高斯低通滤波器处理图像
img = img.*H;

% 对图像进行IDFT
img = ifft2(img);
% 获得图像的实部
```

```
    img = real(img);
    % 将图像中心移回原处
    img = shift(img);
    % 截取图像的左上象限
    img = cut(img);
    figure
    % 显示图像
    imshow(uint8(img));
    % imshow(img,[]);

    end

    function [img] = fill(img)

    [m, n] = size(img);
    % img = [img, zeros(m, n)];
    % img = [img; zeros(m, 2*n)];
    temp = zeros(m*2, n*2);
    temp(1:m, 1:n) = img;
    img = temp;

    end

    function [img] = cut(img)

    [m, n] = size(img);
    img = img(1:m/2, :);
    img = img(:, 1:n/2);

    end

    function [img] = shift(img)

    [m, n] = size(img);
    for i = 1:1:m
        for j = 1:1:n
        img(i, j) = (-1)^(i+j-2)*img(i, j);
        end
    end

    end

    function [H] = GLPF(img, D0)

    [P, Q] = size(img);
    H = zeros(P,Q);
    const = 2*D0*D0;

    for i = 1:1:P
        for j = 1:1:Q
            H(i, j) = exp(-distance(i, j, P, Q)/const);
        end
    end

    end

```

```
80      function [H] = GHPF(img, D0)
81
82      [P, Q] = size(img);
83      H = ones(P,Q) - GLPF(img, D0);
84
85      end
86
87      function [D] = distance(u, v, P, Q)
88
89      D = (u-P/2)^2 + (v-Q/2)^2;
90      D = double(D);
91
92      end
```