电 子 科 技 大 学

实　验　报　告

学生姓名：关文聪　　学 号：2016060601008　　指导教师：徐行

一、 实验项目名称：

Image Filtering and Hybrid Images（图像过滤和混合图像）

二、 实验原理：

图像的频率：

图像的频率指的是空间频率，它和我们认知的物理频率是不同的，因此，要理解图像频率，就要将两者的定义脱离开。图像可以看成是一个定义在二维平面上的信号，该信号的幅值对应像素的灰度（彩色图像对应 RGB 三个分量）。如果仅仅考虑一帧图像的某一行像素，那么，可以看成是一维空间的信号。这种信号和我们常见的时域信号是很相似的，只不过时域信号是定义在时间域上的，而图像信号是定义在空间域的。因此，图像的频率被称为空间频率，空间频率是指单位长度内亮度做周期性变化的次数，它反映了图像的像素灰度在空间中的变化情况，从傅里叶频谱上可以看到明暗不一的亮点，反映的就是某点与邻域间的差异程度。举个例子，一帧图像的背景或者变化缓慢的区域，也就是灰度值分布比较平坦，那么，低频分量就比较强。图像的边缘、细节以及噪声的像素灰度在空间的变化非常剧烈。因此为高频分量。

滤波：

在图像处理或者计算机视觉应用中，在正式对图像进行分析处理前一般需要一个预处理的过程。预处理是对图像作一些诸如降维、降噪的操作，主要是为后续处理提供一个体积合适的、只包含所需信息的图像。这里通常会用到一些滤波处理手法。滤波，实际上是信号处理里的一个概念，而图像本身也可以看成是一个二维的信号，其中像素点灰度值的高低代表信号的强弱。对应的高低频的意义：

高频：图像中灰度变化剧烈的点，一般是图像轮廓或者是噪声。

低频：图像中平坦的，灰度变化不大的点，图像中的大部分区域。

滤波器：

根据图像的高频与低频的特征，我们可以设计相应的高通与低通滤波器，高通滤波可以检测图像中尖锐、变化明显的地方；低通滤波可以让图像变得光滑，滤除图像中的噪声。顾名思义，高通滤波器为：让高频信息通过，过滤低频信息；低通滤波相反。

理想的低通滤波器模板为：

$$H(u,v)=\begin{cases}1, & D(u,v)\le D_0\\ 0, & D(u,v)>D_0\end{cases}$$

其中，D0 表示通带半径，D(u，v)是到频谱中心的距离（欧式距离），计算公式如下：

$$D(u,v)=\sqrt{(u-M/2)^2+(v-N/2)^2}$$

M 和 N 表示频谱图像的大小，（M/2，N/2）即为频谱中心

理想的高通滤波器与此相反，1 减去低通滤波模板即可。

Guassian 滤波：

Guassian 滤波是一种线性平滑滤波，适用于消除高斯噪声，广泛应用于图像处理的减噪过程。通俗的讲，Guassian 滤波就是对整幅图像进行加权平均的过程，每一个像素点的值，都由其本身和邻域内的其他像素值经过加权平均后得到。Guassian 滤波的具体操作是：用一个模板（或称卷积、掩模）扫描图像中的每一个像素，用模板确定的邻域内像素的加权平均灰度值去替代模板中心像素点的值。Guassian 低通滤波器函数为：

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

1 减去低通滤波模板即可得到高通滤波模板

三、实验目的：

对不同图像分别进行高通和低通滤波，融合图片。

四、实验内容：

1. 了解、学习图像滤波操作与图像融合的相关知识。

2. 图像滤波：完善、填充代码，使用滤波器对给定的图像进行滤波处理，分别得到图像的高频与低频信号。

3. 图像融合：利用上述步骤滤波得到的图像高低频信号进行图像融合处理，并输出融合后的图像。

五、实验步骤：

Part 1：图像滤波：

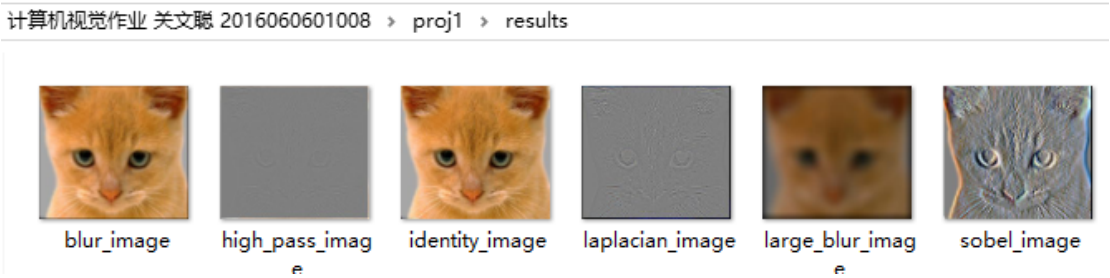
对于此部分，只需要修改完善“helpers.py”文件中的“my_imfilter”函数即可。该函数接收图像输入与 filter，主要实现图像的 padding 操作与卷积操作，但是要注意保持图像尺寸的一致。经过修改完善后的代码如下：

```
1.  def my_imfilter(image, filter):
2.      """
3.      Your function should meet the requirements laid out on the project webpa
    ge.
4.      Apply a filter to an image. Return the filtered image.
5.      Inputs:
6.      - image -> numpy nd-array of dim (m, n, c)
7.      - filter -> numpy nd-array of odd dim (k, l)
8.      Returns
9.      - filtered_image -> numpy nd-array of dim (m, n, c)
10.     Errors if:
11.     - filter has any even dimension -> raise an Exception with a suitable er
    ror message.
12.     """
13.     filtered_image = np.asarray([0])
14.
15.     ##################
16.     # Your code here #
17.     pad_x = (filter.shape[0] - 1) // 2
```

```
    pad_y = (filter.shape[1] - 1) // 2
    # 由于要输出原尺寸图像，因此先对图像进行 padding 操作，再卷积
    image_pad = np.pad(image, (
        (pad_x, pad_x),
        (pad_y, pad_y),
        (0, 0)), 'constant')
    # 计算滤波后图像的尺寸
    filtered_image_height = image_pad.shape[0] - filter.shape[0] + 1
    filtered_image_width = image_pad.shape[1] - filter.shape[1] + 1
    filtered_image = np.zeros([filtered_image_height, filtered_image_width, image.shape[2]])
    # 进行卷积操作
    for k in range(image.shape[2]):
        for i in range(filtered_image_height):
            for j in range(filtered_image_width):
                filtered_image[i, j, k] = np.sum(
                    np.multiply(image_pad[i:i + filter.shape[0], j:j + filter.shape[1], k], filter))
    # raise NotImplementedError('my_imfilter function in helpers.py needs to be implemented')
    ##################

    return filtered_image
```

运行“proj_part1.py”，则会生成文件夹“results”，并输出经过不同滤波操作处理后的若干图片，如图所示：



Part2：图像融合

对于此部分，需要修改完善“helpers.py”文件中的“gen_hybrid_image”函数。该函数中已实现了高斯滤波核函数，函数接收两个图像输入，对输入的图像 1 进行滤波处理得到低频信息，对输入的图像 2 进行滤波处理得到高频信息，最终，将得到的低频信息与高频信息相加，即组合成融合图像。经过修改完善后的代码如下：

```
def gen_hybrid_image(image1, image2, cutoff_frequency):
    """
     Inputs:
     - image1 -> The image from which to take the low frequencies.
     - image2 -> The image from which to take the high frequencies.
     - cutoff_frequency -> The standard deviation, in pixels, of the Gaussian
                           blur that will remove high frequencies.

     Task:
     - Use my_imfilter to create 'low_frequencies' and 'high_frequencies'.
     - Combine them to create 'hybrid_image'.
    """

    assert image1.shape[0] == image2.shape[0]
    assert image1.shape[1] == image2.shape[1]
    assert image1.shape[2] == image2.shape[2]

    # Steps:
    # (1) Remove the high frequencies from image1 by blurring it. The amount of
    #     blur that works best will vary with different image pairs
    # generate a 1x(2k+1) gaussian kernel with mean=0 and sigma = s, see https://stackoverflow.com/questions/17190649/how-to-obtain-a-gaussian-filter-in-python
    s, k = cutoff_frequency, cutoff_frequency * 2
    probs = np.asarray([exp(-z * z / (2 * s * s)) / sqrt(2 * pi * s * s) for z in range(-k, k + 1)], dtype=np.float32)
    kernel = np.outer(probs, probs)

    # Your code here:
    low_frequencies = my_imfilter(image1, kernel)  # Replace with your implementation

    # (2) Remove the low frequencies from image2. The easiest way to do this is to
    #     subtract a blurred version of image2 from the original version of image2.
    #     This will give you an image centered at zero with negative values.

    # Your code here #
    # 用 1 减去低频滤波得到高频滤波
    high_frequencies = image2 - my_imfilter(image2, kernel)  # Replace with your implementation
```

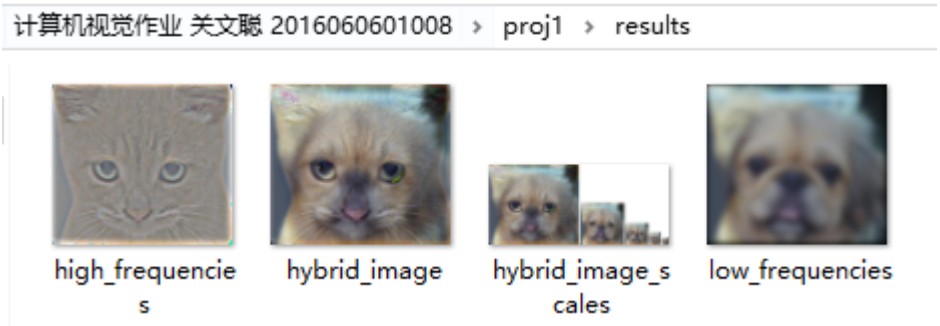
```
35.
36.     # (3) Combine the high frequencies and low frequencies
37.     # Your code here #
38.     # 图像融合=低频 + 高频
39.     hybrid_image = low_frequencies + high_frequencies  # Replace with your i
    mplementation
40.
41.     # (4) At this point, you need to be aware that values larger than 1.0
42.     # or less than 0.0 may cause issues in the functions in Python for savin
    g
43.     # images to disk. These are called in proj1_part2 after the call to
44.     # gen_hybrid_image().
45.     # One option is to clip (also called clamp) all values below 0.0 to 0.0,

46.     # and all values larger than 1.0 to 1.0.
47.
48.     return low_frequencies, high_frequencies, hybrid_image
```

此外，实验过程中，在保存输出图像时，遇到了输出图像全为黑色的问题。经过调试，发现是图像像素存储格式前后不一致造成的。图像的像素有两种表示方式：一种是以 0-1 之间的浮点数表示，另一种则是以 0-255 之间的整数表示。在程序运行过程中，是采用前者的方式进行运算，但存储成文件输出时，应该要采用后者才能正确输出图像，因此，需要进行类型转换，故对文件“proj_part2.py”的“save_image”语句进行如下修改：

```
1.  # save_image('../results/low_frequencies.jpg', low_frequencies)
2.  # save_image('../results/high_frequencies.jpg', high_frequencies+0.5)
3.  # save_image('../results/hybrid_image.jpg', hybrid_image)
4.  # save_image('../results/hybrid_image_scales.jpg', vis)
5.
6.  # 注：经过处理后的图像像素表示形式是 0-1 之间的浮点数，需要转换成 0-255 的整型数才能存
    储为正常的图片，否则所有图片会存储为纯黑色（像素为 0）
7.  save_image('../results/low_frequencies.jpg', (low_frequencies * 255).astype(
    np.uint8))
8.  save_image('../results/high_frequencies.jpg', ((high_frequencies + 0.5) * 25
    5).astype(np.uint8))
9.  save_image('../results/hybrid_image.jpg', (hybrid_image * 255).astype(np.uin
    t8))
10. save_image('../results/hybrid_image_scales.jpg', (vis * 255).astype(np.uint8
    ))
```

运行修改后的“proj_part2.py”文件，在“results”文件夹，分别输出了图像 1 的低频信息、图像 2 的高频信息、融合后的图像、以及对融合图像的缩放对比图，如图所示：



六、实验数据及结果分析：

Part 1：图像滤波：

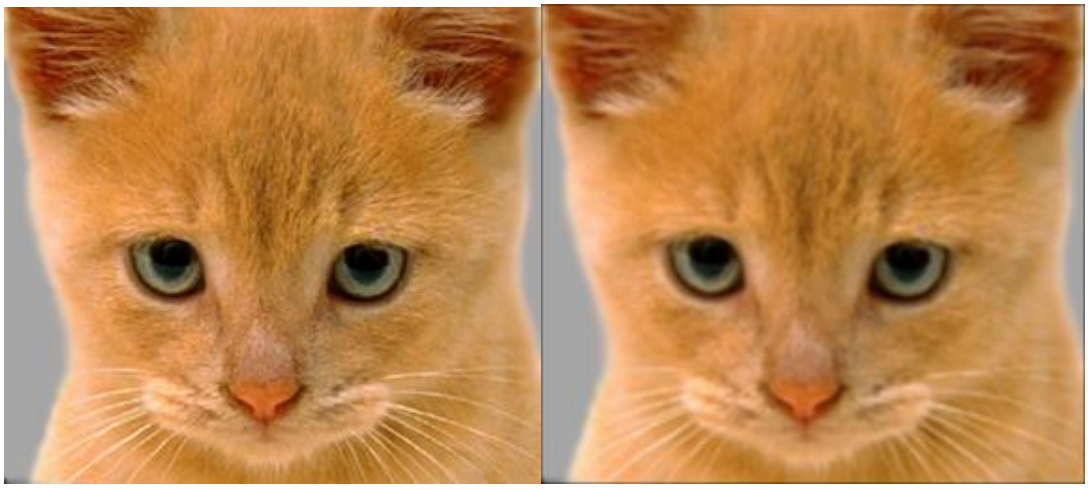


图 1：Identity filter 滤波处理后的图像

图 2：Small blur with a box filter 滤波处理后的图像

图 3：Gaussian blur 滤波处理后的图像

图 4：Oriented filter (Sobel operator)滤波处理后的图像

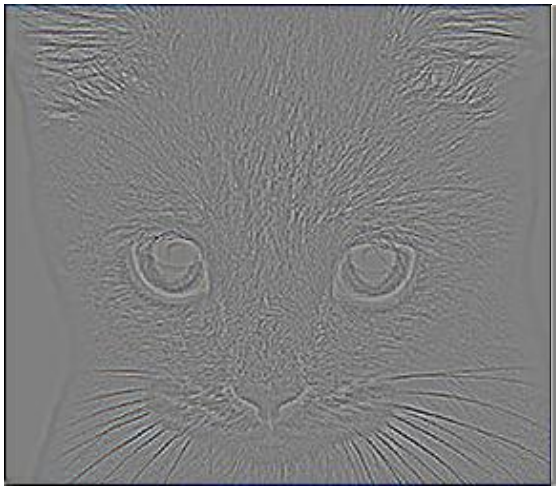


图 5：Laplacian filter 滤波处理后的图像

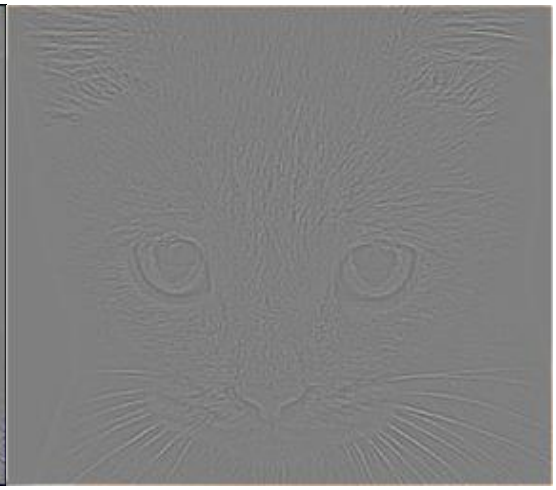


图 6：High pass filter 滤波处理后的图像

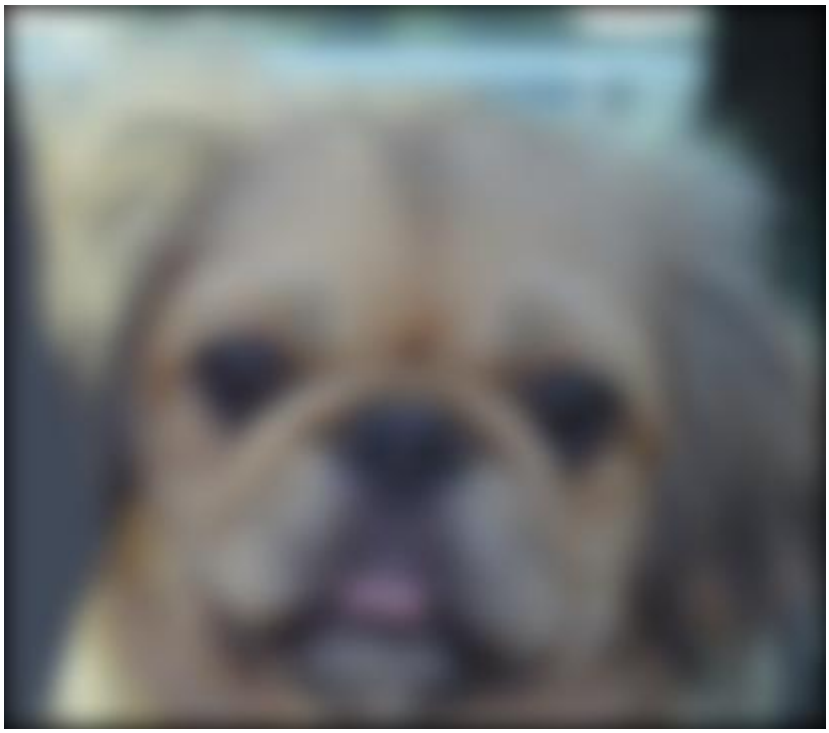
从以上不同输出结果可以看出，不同的滤波器有着各自的特点，即使是对于同一张图片，采用不同的滤波器对其进行滤波处理，得到的结果也是各自不相同的，有些甚至差别非常大。在实际应用时，要根据不同滤波器的特点，选择使用合适的滤波器进行操作。

其中，Identity filter 什么也不做，得到的结果与原图保持一致；Small blur with a box filter 去除了一部分的高频信息，使得结果会比原图稍微模糊；Gaussian blur（高斯模糊）让图像与正态分布做卷积，减少图像噪声以及降低细节层次，模糊效果更好更平滑；Oriented filter (Sobel operator)能很好的保留图像的边界信息；Laplacian filter 可以用于图像增强和空间锐化；High pass filter 可以过滤低频信息保留高频信息，可用于图像锐化。

Part2：图像融合

需要融合的原始图像如下：

其中，dog 图像作为 image1，提取其低频信息，结果如下图：



同理，cat 图像作为 image2，提取其高频信息，结果如下图：

进行图像融合处理后的得到的融合图像如下图所示：

对融合图像进行缩放处理的对比图如下图所示：

可以看出，融合了不同图片高低频信息后的图像，在图像尺寸大小不同时，在视觉上会呈现出不同的感受。在上图就可以很明显直观的看出，当图像尺寸较大，或是观察距离很近时，图像看上去更像是一只猫。而当图像尺寸较小，或是观察距离很远时，图像看上去更像是一只狗。

七、实验结论：

对图像应用不同的滤波器可以实现各种不一样的效果。例如应用 Identity filter 得到原图；应用 Small blur with a box filter 去除部分高频信息，使结果比原图稍模糊；应用 Gaussian blur（高斯模糊）减少图像噪声，降低细节层次，得到更平滑的模糊图像；应用 Oriented filter (Sobel operator)可以保留图像边缘信息;应用 Laplacian filter 进行图像增强和空间锐化;应用 High pass filter 过滤低频信息保留高频信息进行图像锐化等等。

融合不同图像高低频信息得到的融合图像，在图像尺寸不一致或观察距离变化时产生不一样的视觉效果。在图像尺寸较大或观察距离很近时，图像更倾向于呈现出高频信息，而在图像尺寸较小或观察距离较远时，图像更倾向于呈现出低频信息。

八、总结及心得体会：

通过课程的学习和本次实验操作，了解了图像的信息和图像的特征，了解了图像的滤波、融合等操作流程。通过动手操作，锻炼了动手能力，也对课堂上所学知识有了更深刻的理解和认识。

九、对本实验过程及方法的改进建议：

可以提供更详细的文档和实验指导，例如实验指导书等。

报告评分：

指导教师签字：