实验一 :混合图像

从很近的地方看图像，然后再从很远的地方看。你看到了什么？

0. 关键信息

本实验 1 人 1 组完成（即每人要单独完成此实验）。

代码与报告提交信箱：visionexp@126.com

截止日期：2019 年 10 月 8 日 23 点 59 分

1. 概述

实验目标是编写一个图像滤波函数，并用它基于 Oliva、Torralba 和 Schyns 在 SIGGRAPH 2006 发表的题为“Hybrid images”的论文的简化版本创建混合图像。混合图像是静态图像，其解释随着观看距离的变化而变化。其基本思想是，高频往往在感知中占主导地位，但在远处，只能看到信号的低频(平滑)部分。通过将一幅图像的高频部分与另一幅图像的低频部分混合，可以得到一幅混合图像，在不同的距离产生不同的解释。你将使用你自己的解决方案来创建你自己的混合图像。

2. 实施细节

这个题目旨在让你熟悉 Python、NumPy 和图像滤波。一旦创建了图像滤波功能，构建混合图像就相对简单了。

这个项目需要你实现 5 个函数，每个函数都建立在前面函数的基础上:

1) cross_correlation_2d
2) convolve_2d
3) gaussian_blur_kernel_2d
4) low_pass
5) high_pass

2.1 图像滤波

图像滤波(或卷积)是一种基本的图像处理工具。请参阅《计算机视觉——算法与应用》的第 3.2 节和课件材料（第 1 讲：图像和滤波），了解图像滤波(特别是线性滤波)。Numpy 和 openCV 有许多内置和高效的函数来执行图像滤波，但是你却不能直接调用系统提供的函数，而必须从头开始编写自己的函数，更具体地说，你要先实现 cross_correlation_2d（互相关）函数，然后再使用 cross_correlation_2d 实现 convolve_2d（卷积）函数。

互相关的含义：将图像记为 F, 以 H 表示权重核（大小为(2K+1)*(2K+1)），并记 G 为输出图像，则

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i+u, j+v]$$

卷积的含义与互相关近似, 只是权重核“翻转”(水平和垂直)，

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i-u, j-v]$$

在互相关或卷积时，若所用像素位于图像以外，其值设置为 0。

2.2 高斯模糊

正如你在第 1 讲中所看到的，有几种不同的方法可以模糊图像，例如取相邻像素的平均值。高斯模糊是相邻像素的一种特殊加权平均，权值与像素离卷积核中心距离的关系可用高斯函数来描述，

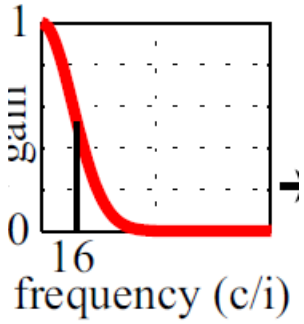
$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

其中 x 为该像素距离卷积核中心的水平距离，y 为该像素距离卷积核中心的垂直距离，σ 为参数（标准差），前面的系数使总和为 1（事实上，你要保证所有权值之和为 1)。

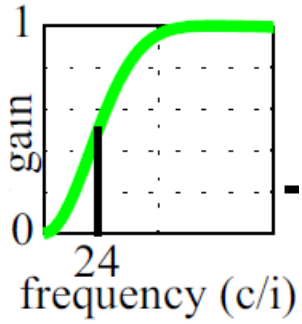
为了实现高斯模糊，你将实现一个函数 gaussian_blur_kernel_2d，该函数产生一个给定高度和宽度的高斯核，之后将会把该高斯核与图像一起传递给 convolve_2d 函数，以产生图像的模糊版本。

2.3 高通和低通滤波器

低通滤波器是从图像(或者实际上，任何信号)中去除精细细节的滤波器，而高通滤波器只保留精细细节，并从图像中去除粗糙细节。



低通滤波器



高通滤波器

可以使用 2.2 节所述的高斯模糊函数，来实现 high_pass 和 low_pass 函数。其中，直接用高斯核与图像卷积，可实现 low_pass 函数。

而高通滤波器在频域可定义为$\mathrm{OutImage} = \mathrm{Image} \cdot (1 - \mathrm{GuassFilter})$。由于卷积满足分配率，因此上式可解释为，先用高斯核与图像卷积得到一个低通图像，然后再从原图中将该低通图像减去，则得到了高通图像。

2.4 混合图像

混合图像是一幅图像的低通滤波版本和第二幅图像的高通滤波版本之和。有一个自由参数，可以针对每个图像对进行调整，该参数控制从第一幅图像中去除多少高频以及在第二幅图像中留下多少低频。这被称为“截止频率”。在 SIGGRAPH 2006 论文中，建议使用两个截止频率(每个图像一个)，你也可以自由尝试。在起始代码中，截止频率通过改变用于构建混合图像的高斯滤波器的标准差（σ）来控制。我们为你提供了使用上述功能创建混合图像的示例代码。

2.5 禁止的函数

本次实验你被禁止使用任何 Numpy、Scipy、OpenCV 或其他预先实现好的卷积或滤波函数进行滤波。这个限制将在未来的实验中取消，但是本次实验，你应该使用 for loops 或 Numpy 数组函数将卷积核应用于图像中的每个像素。你的大部分代码将位于 cross_correlation_2d 和 gaussian_blur_kernel_2d 两个函数中，其它函数将通过调用这两个函数实现。

3. 所提供的文件

hybrid.py：需要你实现的函数都在这个文件中。
test.py：单元测试文件，实现完后可以用这个文件测试你几个基本函数是否正确。
gui.py：该文件提供了一个 GUI，可以帮助你调试图像滤波算法。我们为你提供了一对需要使用 GUI 对齐的图像。校准代码使用仿射变换将眼睛映射到眼睛，鼻子映射到鼻子，等等，正如你在用户界面上指定的。gui.py 是用于 Python2 的。
gui3.py：作用和 gui.py 一样，但是 gui3.py 是用于 Python3 的。
sample-config.json：create_hybrid_image 函数的示例参数
sample-correspondance.json：gui.py 运行时的示例配置文件
Hybrid images.pdf：本实验原理来自该篇 SigGraph2006 的论文。

4. 实验环境配置

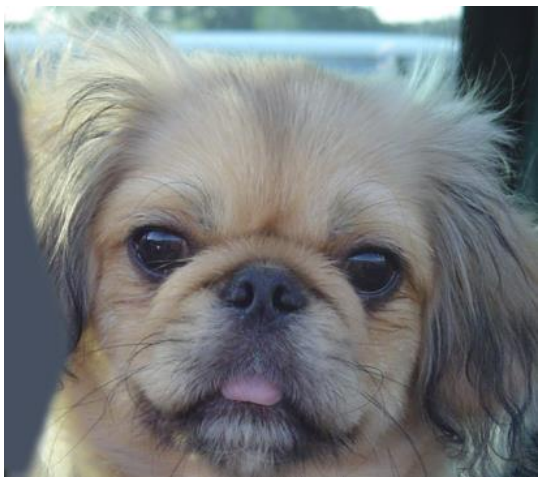
先在 Linux 或 Windows 上安装 Python，之后安装 Numpy 和 OpenCV for python。OpenCV 用于读写图像文件，Numpy 用于实现滤波等函数。如果要运行 gui.py 或 gui3.py，还要安装 PIL（Pillow）。

5. 单元测试

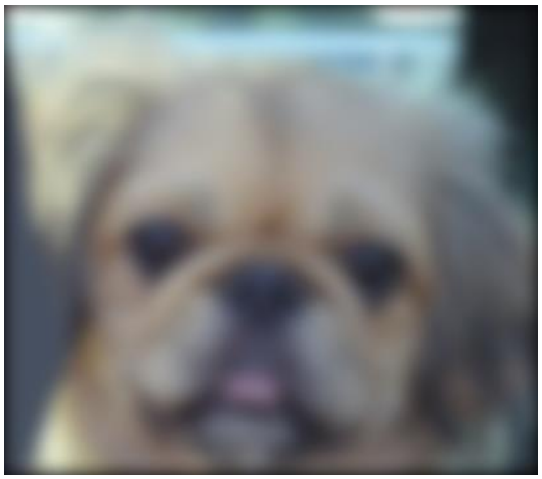
test.py 文件包含了你所实现的几个主要函数的单元测试。直接运行 test.py 会输出单元测试的结果，'.' 表示通过了一个测试，'F' 表示一个测试没过，'E' 表示一个测试出错了（比如你没有实现一个 TODO，raise Exception("TODO in hybrid.py not implemented")语句会产生一个异常错误）。这将允许你以递增的方式测试代码，而无需等到完成所有 TODO 块才去测试。

6. 结果示例

对于本文开始显示的示例，两个原始图像如下所示：




两张图的低通(模糊)和高通版本如下所示:




将高通和低通图像加在一起，你会看到本文开始的图像。如果你看不到图像的多种解释，一种有效的视觉化效果的方法是按如下步骤逐步对混合图像降采样:

我们鼓励你创建其他示例(例如，表情的变化、不同对象之间的变形、随时间的变化等)。有关一些灵感，请参见混合图像项目页面 http://cvcl.mit.edu/hybrid_gallery/gallery.html。

7. 提交文件

hybrid.py：提交所实现的所有五个函数
left.png,right.png：提交用于创建混合图像的左、右图像。(可以是与 gui.py 一起工作的任何格式，不一定是 png)
hybrid.png：提交使用你的程序和左、右图像生成的混合图像
report(.docx,.pdf)：必须说明所使用 Python 的版本、左右图像哪个做高通滤波、哪个做低通滤波，并包含高通和低通滤波器的参数(卷积核大小和标准差 σ)以及混合比。可选地，你可以对你在项目中做的有趣或别具一格的事情写写感想。

将以上文件打包压缩，压缩后的文件命名为“学号+姓名+expX”；如“16030140095 李乐天 exp1.rar”、“16030140095 李乐天 exp1.zip”。Email 的标题类似命名。