课程名称：计算机图形图像技术试验

实验名称：图片放大与超分辨率重建

**姓 名： 应一凡**

**专 业： 智能科学与技术**

**学 号： 21312274**

**实验一 图片放大**

**一、实验内容及目的**

完成常用的插值方法 bilinear、bicubic、nearest 的复现

1. **实验相关原理描述**
2. 最邻近插值法（nearest）

定义：目标各像素点的灰度值代替原图中与其最邻近的像素灰度值

1. 双线性插值法（bilinear）

定义：根据点相邻最近的 4 个点的像素值算出该点的像素

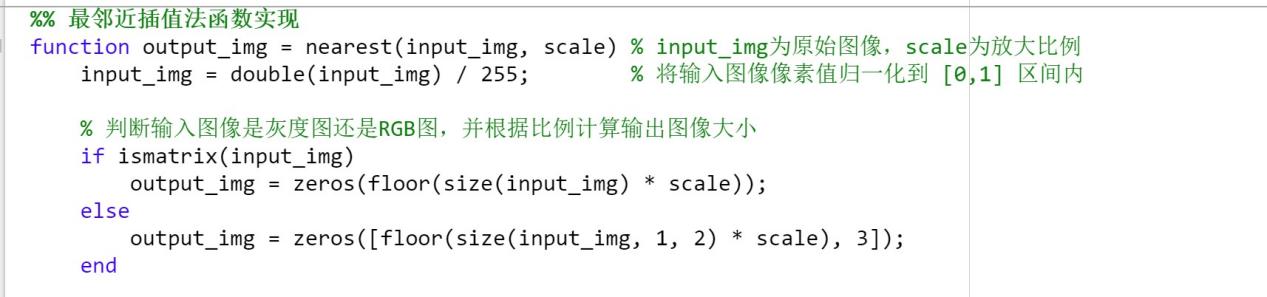
1. 双三次插值法（bicubic）

定义：利用邻近的 4\*4 的点求出权重后插值

**三、实验过程**

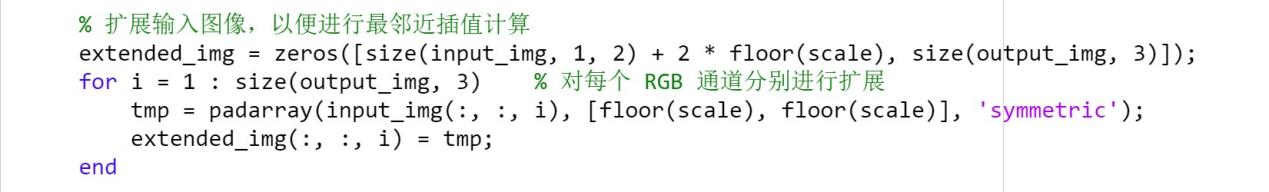
**1.最邻近插值法**

1.1 对输入图像进行预处理并确定输出图像的大小



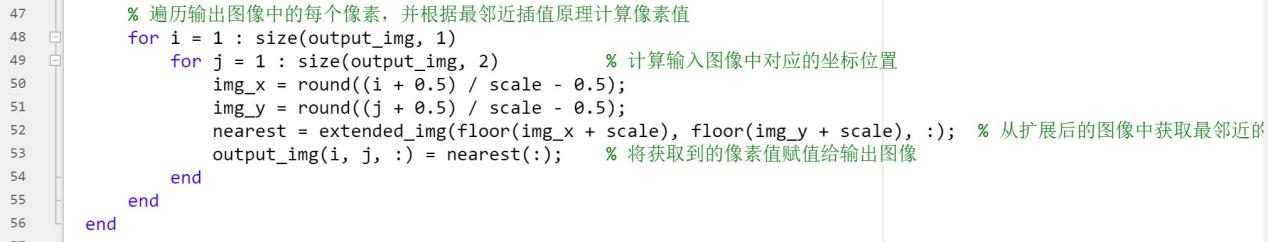
将输入图像的像素值归一化到[0,1]区间内，方便后续的处理，并判断输入图像是灰度图还是RGB图，根据缩放比例计算出输出图像的大小。

1.2 扩展输入图像



最邻近插值需要根据像素点的坐标向最近的整数坐标进行四舍五入，如果不对图像进行扩展，有可能会越界或者计算错误。为了避免这种情况的发生，需要对输入图像进行扩展。在本例中使用 padarray 函数对输入图像的每个通道进行对称填充。

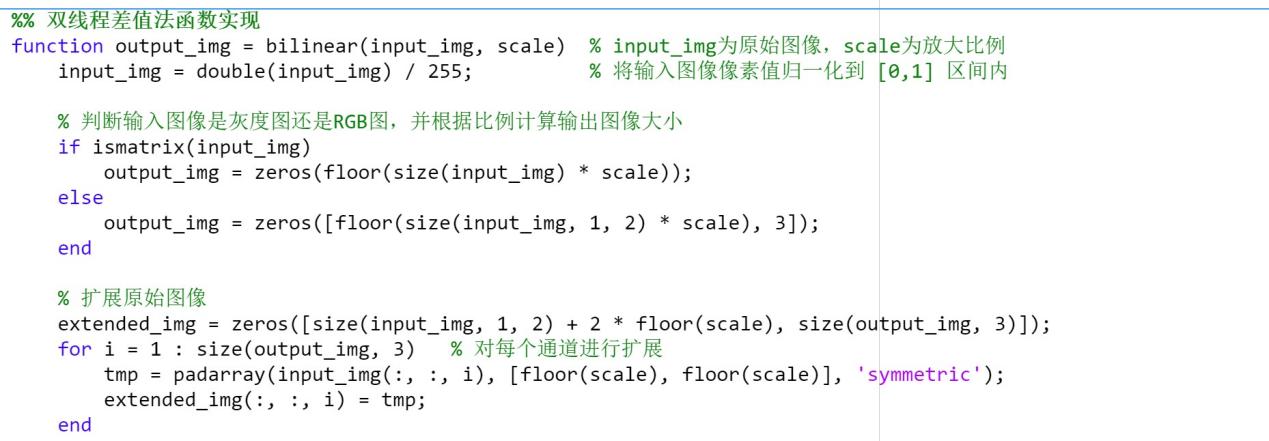
1.3 根据最邻近插值原理计算像素值



首先遍历输出图像中的每个像素。对于每个像素，需要计算其在输入图像中对应的坐标位置，然后根据最邻近插值原理，获取最邻近的像素值作为当前像素的值。其中，i 和 j 分别是当前像素在输出图像中的行列索引（从 1 开始），scale 是缩放比例。由于在计算中需要四舍五入到最近的整数坐标，因此在计算时需要加上 0.5 再向下取整，并减去 0.5 来保证计算结果与上述公式等价。

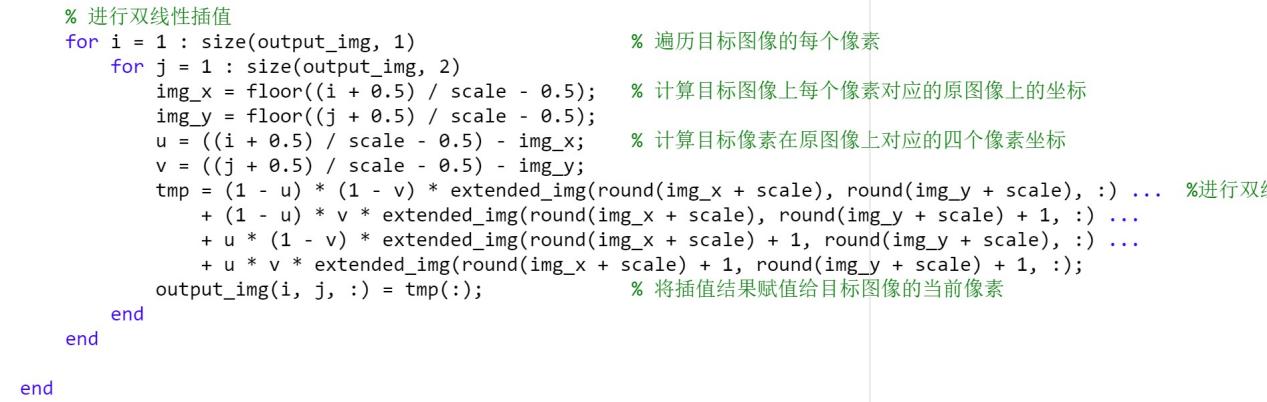
**2.双线性插值法**

2.1 对输入图像进行预处理，确定输出图像的大小并扩展图像



原理同最邻近插值法

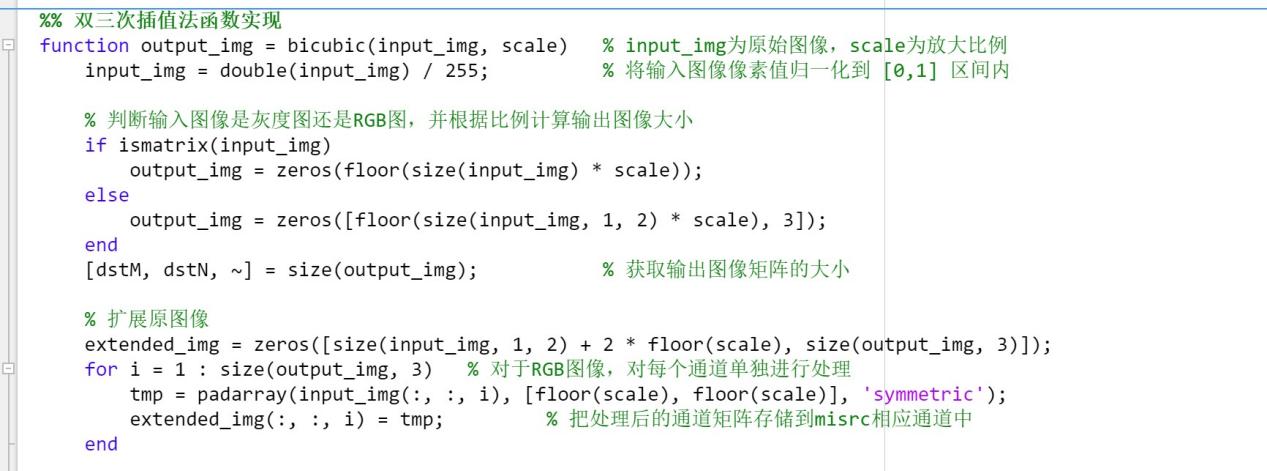
2.2 根据最邻近插值原理计算像素值



通过计算目标像素在其对应四个像素的距离比例，再使用如公式中的权重系数对四个像素进行插值计算，得到当前像素在该通道内的插值结果。最后，将插值结果赋值给目标图像的当前像素，完成双线性插值计算。

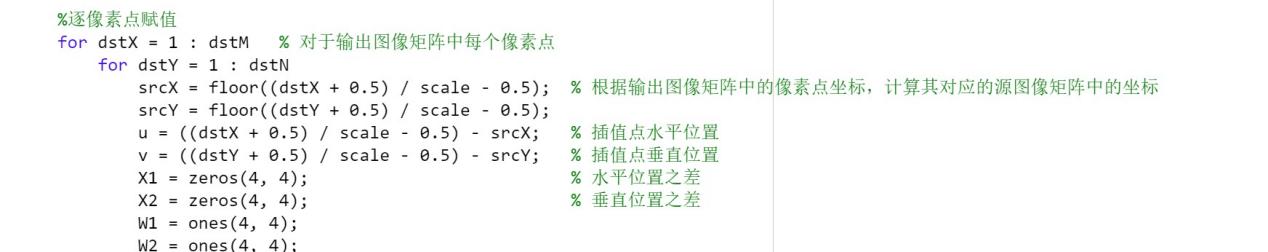
**3.双三次插值法**

3.1 对输入图像进行预处理，确定输出图像的大小并扩展图像



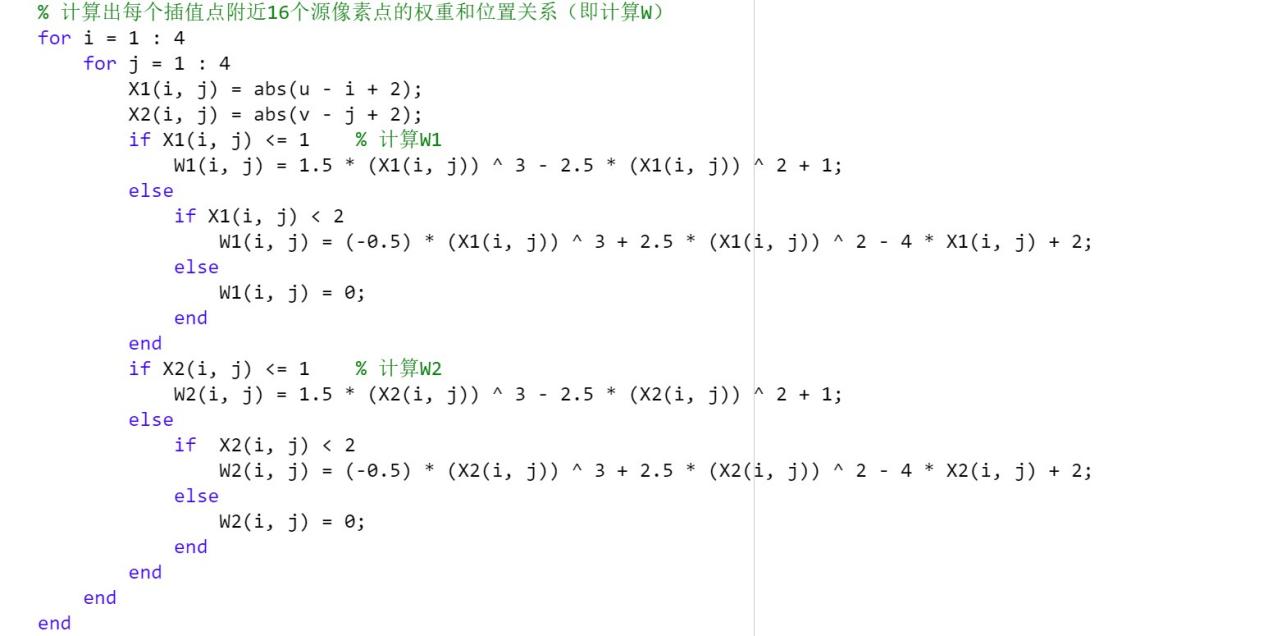
原理同前两个算法的相关部分

3.2 一些量的定义



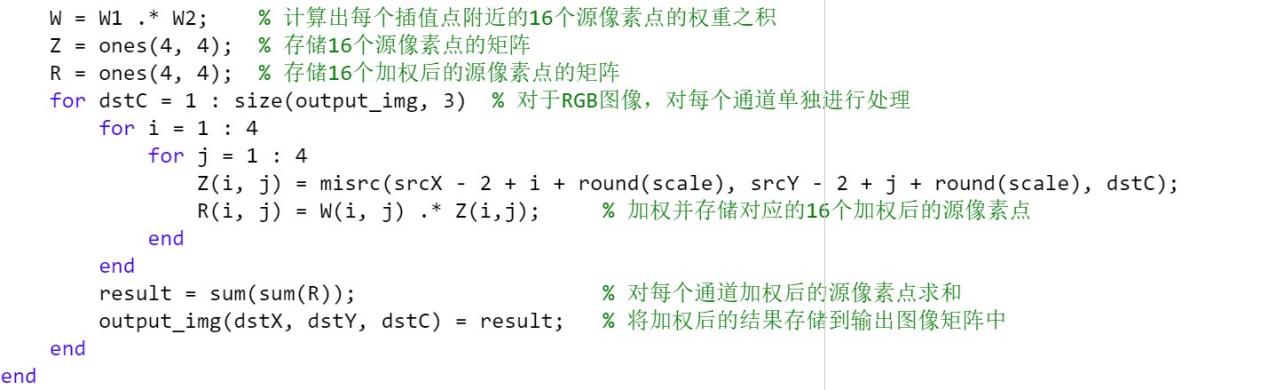
对于每个输出图像矩阵中的像素点，都需要在源图像矩阵中找到与之对应的坐标，然后计算插值点的水平位置和垂直位置，并分别存储在变量 u 和 v 中。接下来，定义两个4\*4的矩阵X1和X2，以及两个4\*4的矩阵W1和W2，用于存储插值点水平位置之差和垂直位置之差，并初始化为全 1。这四个矩阵的作用是在后续的插值计算中用于计算每个像素点四周 16 个像素的权重。

3.3 计算每个相邻点的插值权重



双三次插值算法中，对于每个输出图像矩阵中的像素点，都需要找到其在源图像矩阵中对应的坐标，并计算插值点在源图像中周围相邻点的权重。权重的计算依据位置差 X1 和 X2，通过三次样条插值的公式得到。

3.4 计算每个像素点的插值结果并储存



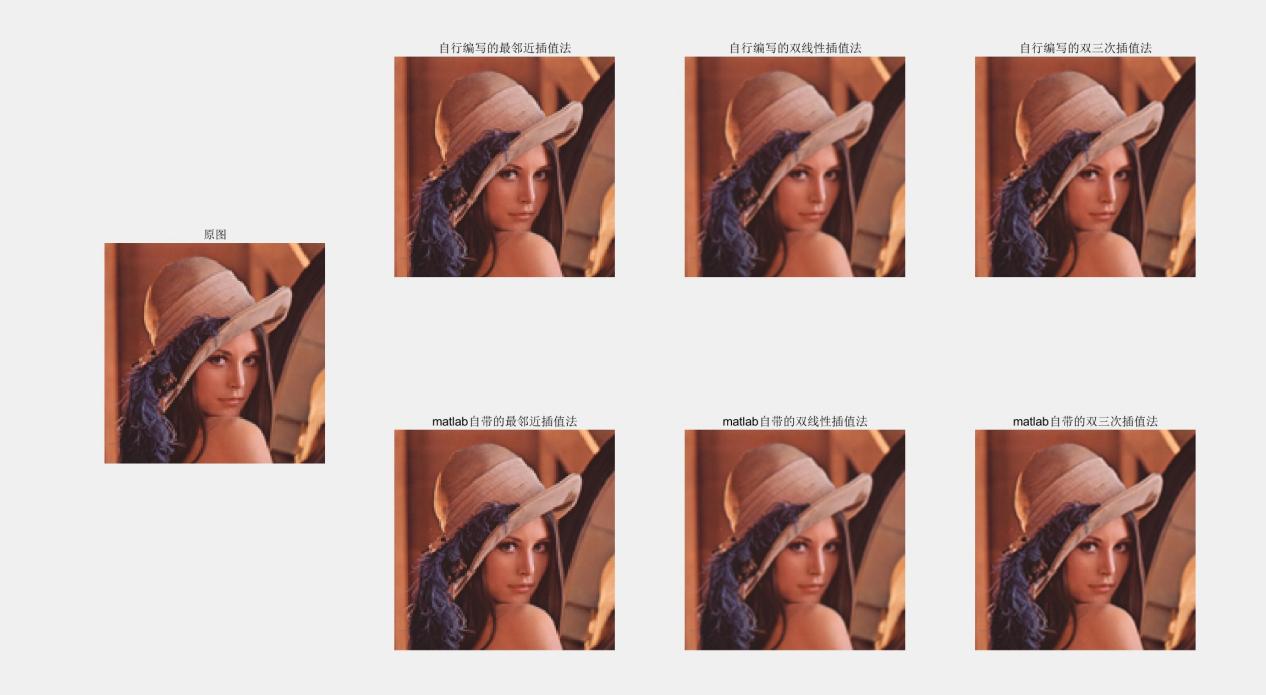
通过将每个相邻点的权重 W1 和 W2 相乘，得到每个插值点附近的16个源像素点的权重之积 W。接下来，对于存储 16 个源像素点的矩阵 Z，遍历每个源像素点，计算其加权后的像素值，并存储在矩阵 R 中。加权过程就是将该像素点的权重 W 与其像素值相乘。

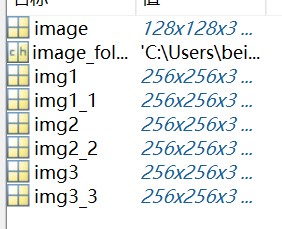
1. **打印结果**

****

打印结果并与matlab自带函数进行对比

**四、实验结果**





由输出结果和工作区变量的图片矩阵大小可知，三种插值结果都符合预期

**五、总结**

1.对于最邻近插值法，其实现简单，但会导致图像出现锯齿状的失真。双线性插值法能够有效地避免锯齿失真，但无法处理弯曲等形变情况。双三次插值法在处理图像缩放时表现最好，它可以在不失真的情况下保持图像细节，并能较好地处理弯曲等形变情况。

2.通过本次实验，我不仅学习到了图像处理中插值算法的相关理论知识，还掌握了如何使用MATLAB实现三种算法，加强了对于该算法的理解。

**实验二 基于字典的超分辨率重建**

1. **实验内容及目的**

完成基于字典的超分辨重建方法

1. **实验相关原理描述**

1）载入字典获取 Dl，Dh，以及低分辨率图 Y，对 Dl 归一化

2）获取特征块大小 patch\_size( sqrt(Dh(size,1)) )，自定义重叠域 overlap(也可以 理解为步长)，超分系数 lambda（这里我们取 0.2）

3）利用插值法，把低分辨率图变大（与目标高分辨率图大小一致），可用 imresize

4）提取 resize 后的低分辨率图特征（有 4 层，1，2 层为一阶导，3，4 层为 2 阶导），与 sobel 算子卷积类似，但是算子格式不一样，4 个算子如下 一阶导算子[-1,0,1] 和 [-1,0,1]T,二阶导算子[1,0,-2,0,1] 和 [1,0,-2,0,1]T

5）对每个特征块求最优高分辨率块（循环）

1. 计算 resize 后低分辨率图像块（5\*5）均值 m

2. 找到对应位置的特征（5\*5\*4）向量，展开为一维向量（100\*1），并且 归一化，得到 Fy

3. 利用 Dl，Fy，求得 A，b，代入函数求得该块的最优稀疏系数 a

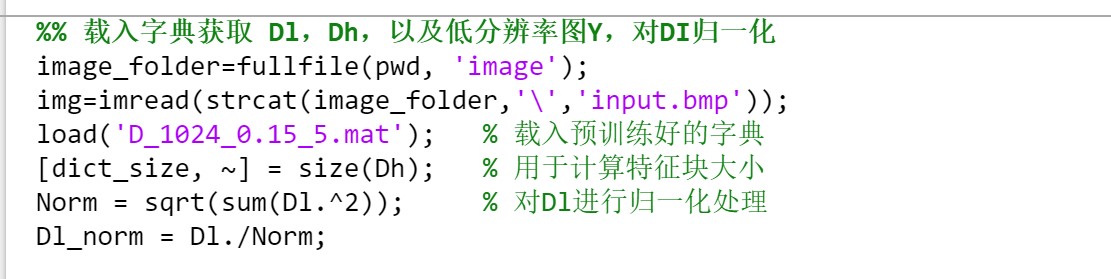
4. 生成高分辨率图块 x = Dh \* a，并将 x+m 加入高分辨率图像 X（相加之 前记得 reshape，保证 Dh 大小与 patch\_size 一致）,且新建一个数组用于记录该 像素格作加法的次数（flag，参数自定）

5. 对像素值的处理，flag 小于 1 的使其等于 1，然后 X./flag （解决过亮问 题）

6. 输出最终图像

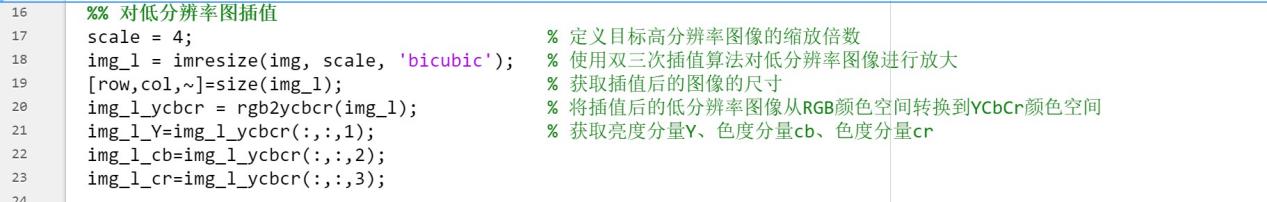
**三、实验过程**

1.载入图片和字典



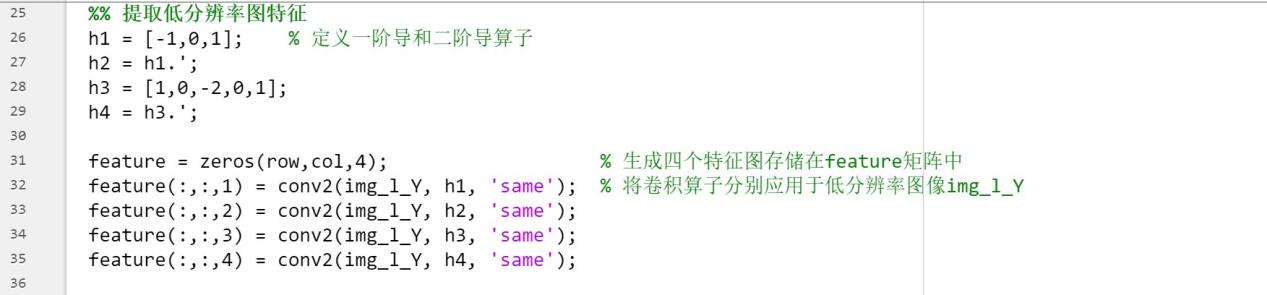
载入一张低分辨率图像，然后加载预训练好的字典，并获取其中的Dl、D，。接着对Dl进行归一化处理。

1. 对低分辨率图像进行插值



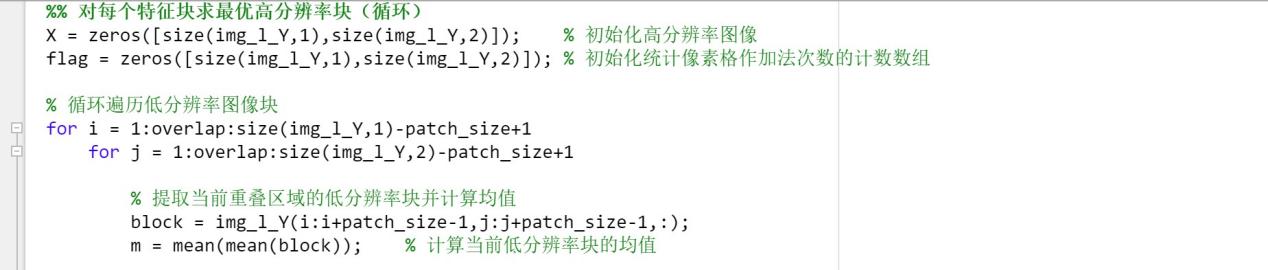
定义了目标高分辨率图像的缩放倍数为scale，然后使用双三次插值算法对低分辨率图像进行放大，将放大后图像的大小保存到row和col中。接着将插值后的低分辨率图像从RGB颜色空间转换到YCbCr颜色空间，并获取其中的Y、cb、cr三个分量，用来输出最后的高分辨率图像。

1. 提取低分辨率图像特征



定义一阶导和二阶导算子，然后生成全零矩阵feature，用于存储四个特征图。分别使用conv2函数将算子h1-h4应用于低分辨率图像img\_l\_Y上，将得到四个特征图，分别存储在feature矩阵的不同通道中。

1. 开始求解最最优高分辨率块的主循环，此部分为计算均值



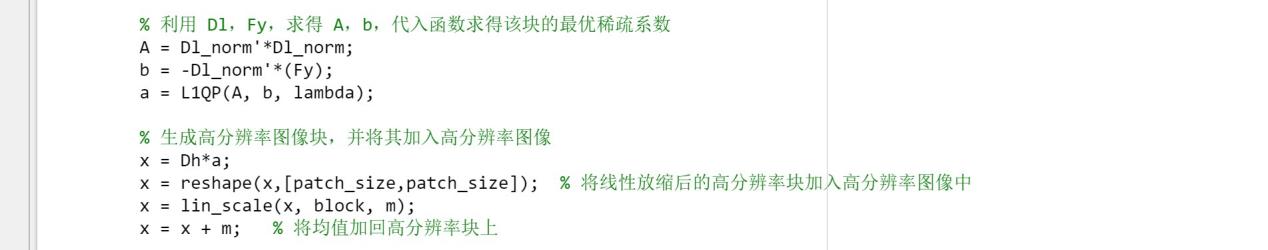
使用双重循环遍历低分辨率图像块，每次取出一个大小为patch\_size\*patch\_size的正方形窗口，其中overlap是指窗口之间的重叠部分大小，用于保证差值重叠部分的合并。对于每个窗口，先提取其中的低分辨率块，计算该块的均值。

1. 获取特征向量展开并归一化



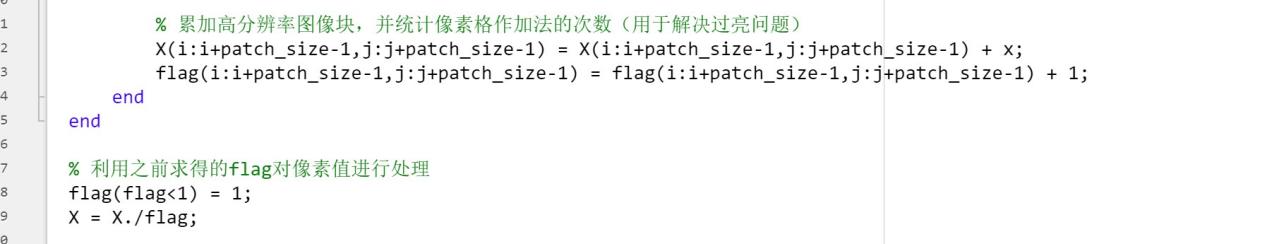
先定义一个全零的行向量feature\_one\_dem，用于存储窗口中的四个特征图的像素值展开后得到的向量。然后使用循环，依次遍历四张特征图的像素点，并将其存储到feature\_one\_dem向量中。其中i和j表示重叠窗口的起始坐标，k表示当前所处理的特征图，用于定位当前像素点。for循环中的代码将当前特征图的像素块展开成一个一维行向量，并将其加入到feature\_one\_dem向量的合适位置中。最后，对feature\_one\_dem进行归一化，得到归一化的特征向量Fy。

1. 求得最优稀疏系数，生成高分辨率图像块，并将其加入高分辨率图像



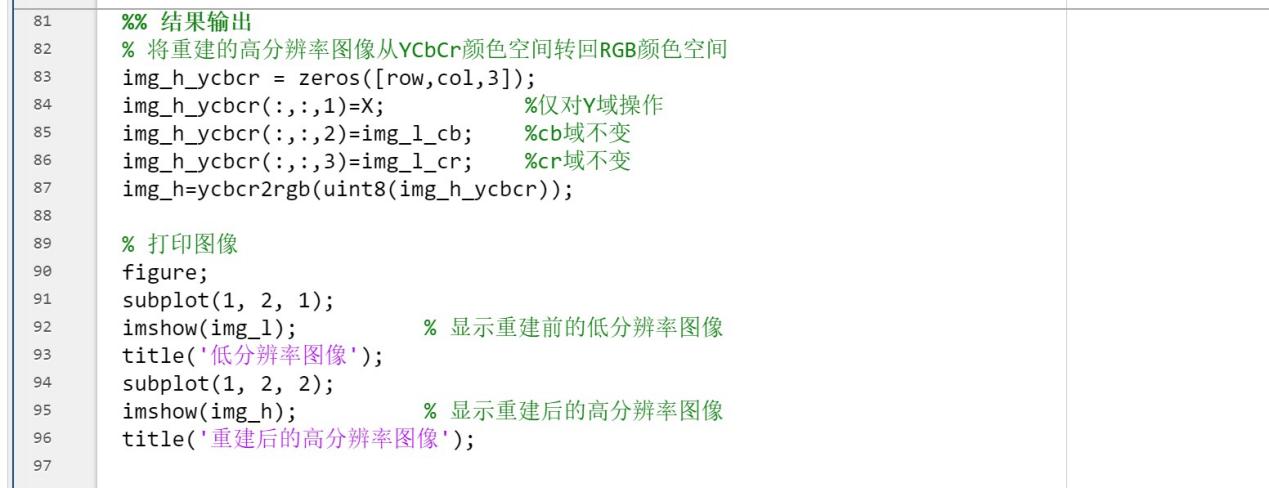
根据字典Dl和当前的特征向量Fy，计算出矩阵A和向量b，并使用L1QP函数求解最优稀疏系数a。接下来，根据求得的稀疏系数a和高分辨率的字典Dh，计算得到高分辨率的像素块x。然后对x进行线性放缩，使其像素值的范围与低分辨率图像块相同，并将均值加回高分辨率块上。最后将该块加入到高分辨率图像中。

1. 累加高分辨率图像块，并统计像素格作加法的次数（用于解决过亮问题）



依次遍历每个重叠块所覆盖的位置。对于每个位置，将其对应的高分辨率图像块x加到X中，并将其对应位置的计数器flag加上1，用于记录该像素格被加多少次。根据flag数组对图像做像素值的处理，以避免出现图像过亮的问题。

1. 结果输出



创建一个空的img\_h\_ycbcr三维数组，用于存储高分辨率图像的YCbCr颜色空间表示。将重建得到的像素值矩阵X存储在img\_h\_ycbcr的第一个通道中，而第二和第三个通道则分别用原始低分辨率图像的Cb和Cr通道填充。然后用ycbcr2rgb转换至rgb类型，最后同时打印低分辨率图和高分辨率图做对比。

**四、实验结果**

****

在已经使用lin\_scale函数的情况下高分辨率图片还是带有一些模糊，可见实验结果并不是令人很满意，考虑是scale（放大倍数）和overlap（步长）设置的影响，此次实验设置scale为4，overlap为2，运行时间15min，如若想得到更清晰的图片，考虑减小步长为1，增大scale，但是由于运行时间过长，无法实践。

1. **总结**
2. 本实验是通过使用稀疏编码的方法对低分辨率图像进行超分辨率重建。在实验中，首先载入预训练好的字典Dl和Dh；然后对低分辨率图像进行插值，并提取其特征进行稀疏编码，最终生成高分辨率图像。
3. 稀疏编码算法可以较好地处理低分辨率图像的超分辨率重建问题。

3. 可以通过调整字典尺寸、步长、放大尺寸等超参数，达到更好的重建效果，但是会带来更高的计算成本。

4. 稀疏编码算法对于一些复杂的场景或图像仍然存在不足，需要更加细致的处理和优化。