专业：信息工程
姓名：徐晓刚
学号：3140102480
日期：2017 /3 /18
地点：

浙江大学实验报告

课程名称：通信原理实验　指导老师：项志宇　成绩：
实验名称：图像插值算法实现　实验类型：探究型　同组学生姓名：无

一、实验目的和要求（必填）　二、实验内容和原理（必填）
三、主要仪器设备（必填）　四、操作方法和实验步骤
五、实验数据记录和处理　六、实验结果与分析（必填）
七、讨论、心得

一.　实验目的和要求

1. 掌握在对图像降采样之后，进行图像插值恢复原来图像大小的插值算法的原理；
2. 需要编程实现三种插值算法，分别是最近邻插值，双线性插值，双三次插值；
3. 比较这三个算法的效果，做出分析。

二.　实验内容和原理

（1） 实验原理

1. 最近邻插值：

原理：当一幅大小为$M \times N$的图像$f(x,y)$（$f(x,y)$在这里指的是在图像坐标(x,y)点处的灰度值）

进行尺度变换，变换之后的图像大小变为$(M \times xscale) \times (N \times yscale)$
$xscale$和$yscal$分别是在x, y方向上图像尺度的变化比例

设置变换之后的图像灰度表示为$g(u,v)$，最近邻插值的原理是对于将目标的像素点的灰度值设置为在原图像中最接近的点的灰度值。如果只是简单的线性尺度变化，那么我们容易得到变换之后的坐标的关系：

$$x = floor(xscale \times u), y = floor(yscale \times v)$$

由此我们可以对目标图像中的每一个点计算出其在原始图像中对应的点的坐标值，并且由此完成尺度变换（放大或者缩小）

2. 双线性内插：

原理：虽然最近邻插值法比较简单，但是最近邻插值法得到的图像一般效果不是很好。而双线性内插法一般在表现上明显好于最近邻插值。双线性内插法需要找寻四个点的信息来进行目标图像中的灰度值的设定。

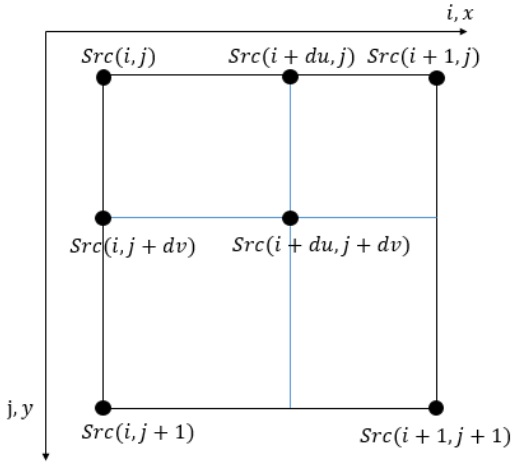
首先我们知道目标图像的坐标和原始图像的坐标之间的关系依旧满足进行尺度变换的比例关系：

$$x = xscale \times u, y = yscale \times v$$

一般来说上述式子得出的坐标值并不是整数，所以我们可以得到以下的式子：

$$i = floor(xscale \times u), j = floor(yscale \times v)$$
$$du = xscale \times u - i, dv = yscale \times v - j$$

下面我们需要画出其示意图进行分析：



我们分别假设在点 $Src(i,j), Src(i+1,j), Src(i,j+1), Src(i+1,j+1)$ 点的灰度值分别为 a,b,c,d 。所谓的双线性内插也就是指的是需要在 i,j 两个方向上实现线性插值。所以依照这些点的值，我们可以得出以下的两个线性关系式：

$$Src(i+du,j)=(b-a)\bullet du+a$$
$$Src(i+du,j+1)=(d-c)\bullet du+c$$

上述的关系式能够继续推倒出我们需要的点 $Src(i+du,j+dv)$ 的灰度值，其计算过程以及结果如下所示：

$$Src(i+du,j+dv)=[Src(i+du,j+1)-Src(i+du,j)]\bullet dv+Src(i+du,j)$$
$$=(1-du)(1-dv)\bullet a+(1-dv)\bullet du\bullet b+(1-du)\bullet dv\bullet c+du\bullet dv\bullet d$$

对于在目标图像中的每一个点，我们用上述式子进行计算，每一个点牵涉到原始图像中的四个点的值。

3. 双三次内插：

原理：在双三次内插中，我们需要使用原始图像中更多的点的信息，并且为每一个像素点估计权重，这样才可以通过原始的像素点来计算变换之后的目标像素点的灰度值。下面假设在双线性内插的讨论中的变量 i,j,du,dv 的含义不变。为计算目标点的像素值，我们需要 16 个点的信息：

$$\text{原始图像中的坐标}x\in\{i-1,i,i+1,i+2\},y\in\{j-1,j,j+1,j+2\}$$

在这里为了得到这 16 个像素点的权重，我采用了基于 Bicubic 基函数的双三次内插法。具体的公式如下说明（假设 $f(i+du,j+dv)$ 为所对应的原始图像中的坐标的灰度值）：

$$f(i+du, j+dv) = ABC^T$$

$$A = [W(du+1), W(du), W(du-1), W(du-2)]$$

$$C = [W(dv+1), W(dv), W(dv-1), W(dv-2)]$$

$$B = f(i-1:i+2, j-1:j+2)$$

其中W为三次插值核函数，可以由如下式子表示：

$$S(x) = \begin{cases} (a+2)|x|^3 - (a+3)|x|^2 + 1 & \text{when } |x| \le 1 \\ a|x|^3 - 5a|x|^2 + 8a|x| - 4a & \text{when} |x| \le 1 \\ 0 & \text{otherwise} \end{cases}$$

由此我们可以得到关于这 16 个像素点的权重，并且计算出目标像素点所对应的灰度值。

4. 降采样原理

图像的降采样就是在图像上依据缩放的比例，在图上依次取点。可以看成是一个采样的过程，只不过是在二维平面上的采样。图像坐标满足的关系如下：

$$x / xscale = u, y / yscale = v$$

x, y分别为缩放后图像的坐标；

u, v分别为缩放前图像的坐标

（2） 实验内容

1. 实现最近邻图像内插算法；
2. 实现双线性图像内插算法；
3. 实现双三次图像内插算法。

三、主要仪器设备

计算机， VS2013 编程环境， Opencv2.4.10

四、操作方法和实验步骤

具体的代码实现采用 VS2013 编程环境，并且使用 Opencv2.4.10 工具包。

1. 首先进行算法原理的阐述，弄清其实现过程，具体的实验原理对于最后的结果有着重要的影响；
2. 对最近邻内插算法，双线性内插算法，双三次内插算法分别进行实现，采用 Opencv2.4.10 中的函数；
3. 对三种算法的效果进行评估，并且提出改进。

其中在一开始需要重点注意的是 IplImage 格式能够读取和表示图像；

图像一开始的降采样过程采用函数 cvResize(img, dst, CV_INTER_NN)来实现。

五、实验数据记录和处理

首先我们选取的测试图像如下图所示：

降采样之后的效果如下：

在这里我设定图像长和宽方向上的缩放，即取样比例都是 0.5，在程序中用 fScale 这个变量来体现。

三种算法实现的效果如下：

1. 最近邻内插算法：

2. 双线性内插算法：

3. 双三次内插算法：

六、实验结果与分析

由最近邻内插算法得到的恢复图像效果并不是很好，存在较多的毛刺和噪声。相比较而言，双线性内插法的效果就变得比较好，不但图像更加清晰，而且没有太多的噪声。双三次内插的效果也比最近邻算法效果要好，但是却比双线性内插算法的效果差。一开始这个效果我感到很困惑，后来重新对进行双三次权值分配的 Bicubic 基函数进行了调整，发现效果有改善。这表面双三次内插的效果与选用的三次函数核有很大的关系。

七、讨论、心得

此次的实验中比较重要的一个部分就是学会了如何使用 Opencv 工具。Opencv 是一个强大的图像编辑和操作工具包，相比于 matlab 来说也有很大的有点。这次的三种插值函数实现过程中，让我对 Opencv 如何操作图像有了基础的了解，能够编程实现。

在算法方面，最近邻插值比较容易实现，但是双线性和双三次插值的算法由于没有现成的公式，而且书上也没有讲，所以只有自己先进行推倒。在实现双三次插值算法的时候遇到了无法推倒的困境，最后在查阅资料和询问学长之后，才决定用 Bicubic 函数进行插值处理，用像素周围的 16 个点来进行最后的计算。虽然最后双三次插值的效果并不比双线性插值的效果要好，但是体现出比最近邻算法要好的效果。这表明我的算法思路基本上正确，但是还是需要再改进。