数值分析与算法大作业 1 实验报告

自 45 柳荫 2014011858

变形函数的选取与设计：

图像旋转：

对于图像旋转，相对来说比较容易实现：
有一个旋转预处理函数：preRotate，其实就是让用户输入一些旋转的参数，包括旋转中心的位置，旋转的半径，中心点的旋转角度（可以正负）等，从而可以实现在不同位置进行不同范围内程度不同的各种旋转。

之后是旋转函数 rotate，代码中注释已经写得很清楚，对于一个圆形区域内的像素点进行旋转映射然后插值，通过课件上的函数，可以比较方便地实现。对于坐标点映射部分，此题并没有通过极坐标，因为 tan 函数使用时会遇到一些没有定义的情况，不太方便，所以直接通过数学关系得到了旋转后点的坐标和原来点的坐标的关系，用正弦、余弦函数就可比较方便地实现。

图像畸变：

对于图像畸变，在网上找到了相应的资料，知道了有桶形畸变和枕形畸变。同时也先预处理了一下，其实主要就是找出图像的中心点，并且让用户输入是放大（桶形）还是缩小（枕形）。

若是桶形畸变，在原来的点至中心点的距离和现在的点距中心点的距离中满足如下关系：

$$K * rd^3 + rd = ru.$$

其中 k 为畸变系数，需要自己调试得到，程序里选择了 0.000005，rd 为某目标像素点对应的原图中的点距离中心点的距离，ru 是目标像素点距离中心点的距离。所以用了递归的方式构造了一个求解方程组的函数 solveFunction，给定 ru，解出 rd，接下来就是插值部分了。

若是枕形畸变，也是类似的，不需要解方程了，按照资料上的公式，将目标点的位置映射回原图像素点（double），再进行插值就行了。

TPS 网格变换：

这题虽说是选做题，看上去比较难，但是实际上只是涉及了一系列的矩阵变换，关键还是要找到目标点与对应的原图点的映射关系。这里我选用了原图中的 5 个点，分别为中心点，然后是横纵坐标分别位于原图 1/5、4/5 的 4 个点，作为控制点，目标控制点是分别随意对控制点做了一些平移以后的点。接着计算课件中的矩阵 L，除了对角线上的元素是 0，其他都是 2 个点之间的径向基函数，这是左上角矩阵 K，再构造出 n*3 的矩阵 P，用 P 和 P^T 补充 L。用目标点构造矩阵 Y，用 L 的逆左乘 Y，得到线性方程组的解，a1,ax,ay,w1~w5，再用下式确定每个点的映射关系。

$$f(x,y)=\left[f_x(x,y),f_y(x,y)\right]^T=\boldsymbol{a}_1+\boldsymbol{a}_x x+\boldsymbol{a}_y y+\sum_{i=1}^{n}\boldsymbol{w}_i U\left(\left|P_i-(x,y)\right|\right)$$

使用插值方法的介绍与分析：

本次程序对必做部分都用了 3 种插值方法，对选做的 TPS 只尝试着用了 openCV 中 Mat 类写了一下 bilinear 插值。

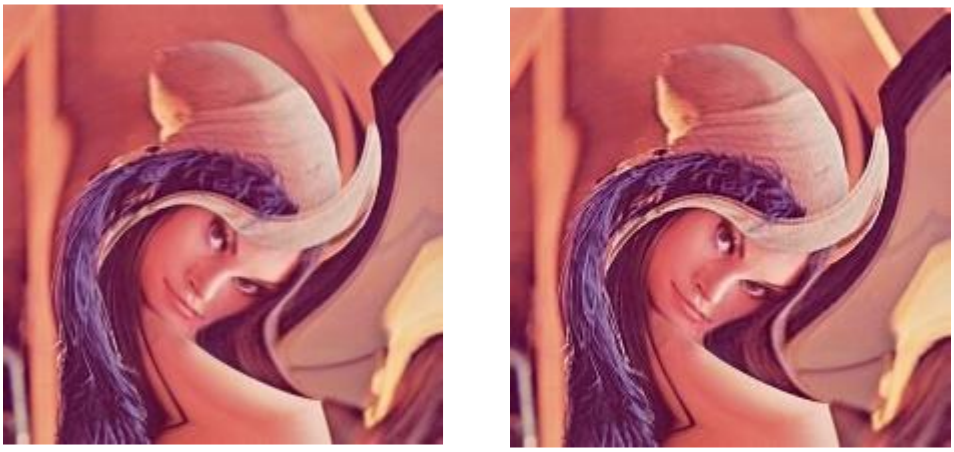
最近邻、双线性、双三次插值的形参都是目标像素点的位置（u, v）与算好的其对应的在原图中的（x, y）,其中 u、v 是整型，x、y 是浮点型。最近邻就是把离(x,y)最近的点的像素值赋给(u,v)，双线性和双三次都是借助公式同样的道理赋值，分别把对应原图像某浮点坐标的整数、小数部分取出来进行相应的矩阵运算。代码注释写的很清楚。由于 rgb 图像是 3 通道，所以任何一个像素点的赋值都要用一个 for 的 3 重循环对其每一个通道分别作插值运算。

对于插值作用效果，双三次速度明显要慢于前两个。

图像旋转：

对于效果来说，后 2 种插值方法要好于第一种，图像毛刺不明显：



上图左上是原图，右上、左下、右下分别是对中间半径 100 的圆中心点旋转-90°最近邻、双线性、双三次插值后的结果，可以看出后两种图像中边缘更圆滑。

对于图像畸变，效果也是这样：

从上到下依次是最近邻、双线性、双三次的枕形畸变插值，仔细看 C 罗的左臂可以看出后 2 张更光滑，毛刺更少。若在照片查看器中依次查看各张图片，盯着一个地方，则更容易看出差别，包括后 2 张的细微差别。

TPS 变形只用 Mat 类写了双线性插值，原图和结果图如下：

由于涉及大量矩阵运算，因此相对来说时间较长，大一些的图像一般处理会超过一分钟。

以上就是插值的方案设计和方案的基本原理。

误差分析：

由于是计算机严谨的科学计算，观测误差不存在，模型误差也可忽略。

所以这次图像处理的误差主要是方法误差和舍入误差。

首先是舍入误差：开始时是对 PI 取的 3.1415926，这存在舍入误差，在计算过程中不断地在整型和浮点型中做计算也会产生大量的舍入误差。

然后是方法误差：在数学模型不能得到精确解时，通过数值方法求近似解会产生截断误差，在这个问题中表现最明显的是图像畸变时求解的那个一元三次函数，$K * rd^3 + rd = ru$.，程序中，用递归的方法求解，某个地方是以 1 递增，所以很难求精确解，智能用一个极小的数 eps 作其限制，这势必会产生方法误差。

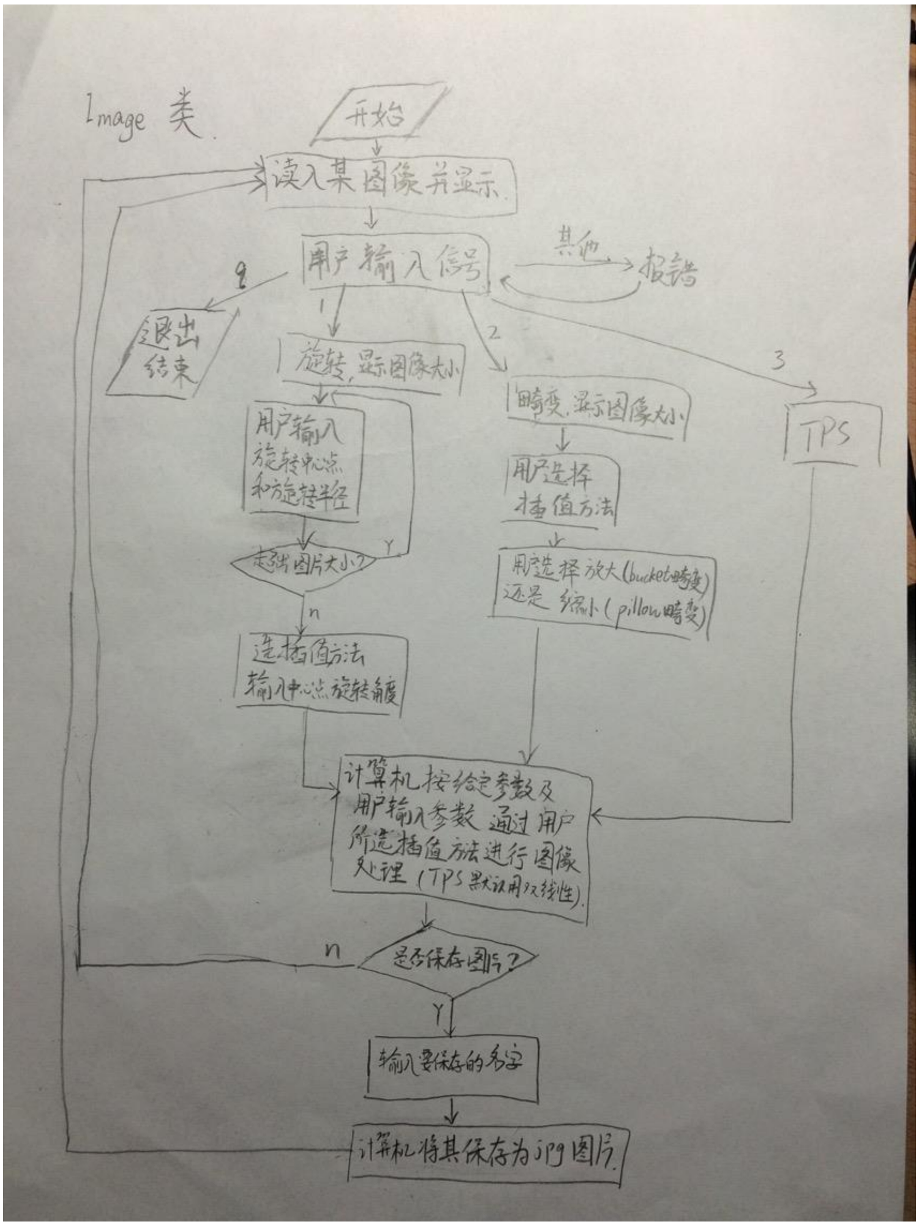
对插值函数的误差做一些分析，如下（假定新图中（u，v）点映射到原图中为（x，y），u，v 整型，x，y 浮点型）：

最近邻插值：对 x 、y 找其最近邻，会引起不超过 $0.5\sqrt{2}$（≈0.7）个像素距离的误差，反映到像素值上，纯粹对其方法来说的话，不存在误差。

双线性插值：用其四围四个点加权赋给 x，y，除对 x，y 的计算时产生的舍入误差外，还存在加权平均时的舍入误差（因为即使计算机的 double 字长也有限）、最后转成 0~255 内整数时的舍入误差等。

双三次插值的误差分析基本同双线性插值。

程序框图：



实验结果与分析的补充：

旋转、畸变的效果还是不错的，TPS 只是在程序中默认选择了一些控制点和目标点，其实也是可以让用户自己输入的，考虑人机交互的方便，就不让用户选择了。实际上只要在程序中将控制点和目标点的坐标改一改也是可行的，就变成了另一种形状的 TPS 变形。

畸变部分参考网址：http://m.blog.csdn.net/article/details?id=52473422