图像扭曲变形

2015011462　董家富　自 54

1 需求分析

本次作业要求编写一个图像扭曲程序，可以实现对图像的旋转变形、水波纹变形、B样条变形。

旋转变形：确定旋转中心、旋转半径和旋转角度之后，在该制定区域完成旋转扭曲。

水波纹变形：确定中心、扭曲半径、参数ϱ和φ之后，在指定区域完成水波纹变形。

B样条变形：交互式给定Nx和Ny（变形范围），鼠标拖动一个点到另一个点。当鼠标左键松开时，自动完成周围区域B样条变形功能。

2 编译环境

Windows 10 教育版系统

Qt Creator 5.7

3 基本原理

3.1 变形原理

3.1.1 图像变换

图像变换的本质是将像素点的坐标通过某一种函数关系，映射到另外的位置。

常用的映射可以分为两类：向前映射和向后映射。

向前映射：

把输入图像的像素一个一个映射到输出图像。输入图像上整数点坐标映射到输出图像之后，变成了非整数点坐标。因此，需要将其像素值按一定权重分配到其周围四个像素点上。对于输出图像而言，其整数点像素值周围会有很多输入图像像素映射过来，每个到其周围的非整数点像素值都会分配一定的灰度值到它上面，将这些分配而来的像素值叠加，就是输出图像整数点位置的像素值。由于这个分配、叠加的特性，向前映射法有时也叫像素移交映射。

向后映射：

把输出图像的像素一个一个映射到输入图像。我们知道输出图像上整数点位置在变换前位于输入图像上的位置，一般来说这是个非整数点位置，利用其周围整数点位置的输入图像像素值进行插值，就得到了该点的像素值。我们遍历输出图像，经过坐标变换、插值两步操作，我们就能将其像素值一个个地计算出来，因此向后映射又叫图像填充映射。

3.1.2 旋转变换

旋转变换的本质是利用图像中的坐标点到旋转中心点的位置不同，旋转不同角度，从而实现旋转扭曲。

其向前映射基本公式如下：

$$\mathrm{x} = \mathrm{r}\cos\left(\alpha + \frac{\theta(R-r)}{R}\right) + x_0$$

$$\mathrm{y} = \mathrm{r}\sin\left(\alpha + \frac{\theta(R-r)}{R}\right) + y_0$$

上式中，r 和α为输入图像中坐标的极坐标表示形式。Θ为旋转角度，R 为旋转半径，x、y 为输出图像中坐标，x_0、y_0为输入图像中坐标。观察该式可以发现，距离旋转中心越远的坐标，其 r 就越大，旋转的角度也就越大。这也是产生旋转效果的本质。

在实现旋转扭曲时，我是采用的向后映射，根据该变换公式求出了其逆变换公式为：

$$x_0 = \mathrm{rcos}\left(\alpha - \frac{\theta(R-r)}{R}\right) + x$$

$$y_0 = \mathrm{rsin}\left(\alpha - \frac{\theta(R-r)}{R}\right) + y$$

此时，r 和α为输出图像中坐标的极坐标表示形式。Θ为旋转角度，R 为旋转半径，x、y 为输出图像中坐标，x_0、y_0为输入图像中坐标。

3.1.3 水波纹变形

水波纹的本质依旧是利用点到中心位置的距离不同，旋转不同的角度来实现扭曲。只不过此时旋转角度改变了，加入了正弦变换，利用正弦变换的周期性，实现水波纹周期波动的效果。

其向前映射公式如下：

$$\mathrm{x} = \mathrm{r}\cos(\alpha + A\sin(\frac{\rho}{R} * r + \varphi)) + \ x_0$$

$$\mathrm{y} = \mathrm{r}\sin(\alpha + A\sin(\frac{\rho}{R} * r + \varphi)) + \ y_0$$

其中，r和α为输入图像中坐标的极坐标表示形式。A为水波纹振幅系数，ϱ和φ为水波纹周期和相位因子，x_0、y_0为输入图像中坐标。观察该式子，可以发现，距离旋转中心不同的点，其旋转角度也是不同的，只是不再与r呈线性递减的关系，而是正弦关系。调节参数A, ϱ，φ即可调节水波纹的振幅、频率和相位。

3.1.4 B 样条变形

给定 m+n+1 个平面或空间顶点$P_i(i = 0,1,2, \ldots, m+n)$，称为 n 次参数曲线段。

$$P_{k,n}(t) = \sum_{i=0}^{n} P_{i+k} G_{i,n}(t) \ , \quad t \in [0,1]$$

为第 k 段 n 次 B 样条曲线段(k=0,1,2,…,m)，这些曲线段全体称为 n 次 B 样条曲线，其顶点$P_i(i=0,1,2,\dots,m+n)$所组成的多边形称为 B 样条曲线的特征多边形。其中$G_{i,n}(t)$称为基函数，$P_i(i=0,1,2,\dots,m+n)$称为控制点。

我采用的是三次 B 样条变形，所以其基函数为：

$$\begin{cases} G_{0,3}(t)=\dfrac{(1-t)^3}{6} \\ G_{1,3}(t)=\dfrac{3t^3-6t^2+4}{6} \\ G_{2,3}(t)=\dfrac{3t^3+3t^2+3t+1}{6} \\ G_{3,3}(t)=\dfrac{t^3}{6} \end{cases}$$

其中$t\in[0,1]$，若某个控制点P_i移动了ΔP_i，区间长度$u_{j+1}-u_j=N_x$，则点 x 处的位移为：

$$v(x)=\sum_{l=0}^{3}G_{l,3}(u)\Delta P_{i+l}$$

其中，$u=\frac{x}{N_x}-\left\lfloor\frac{x}{N_x}\right\rfloor$，$i=\left\lfloor\frac{x}{N_x}\right\rfloor-1$。

将其扩展到二维，有

$$v_x(x,y)=\sum_{l=0}^{3}\sum_{m=0}^{3}G_{l,3}(u)G_{m,3}(v)\Delta P_{x(i+l,j+m)}$$

$$v_y(x,y)=\sum_{l=0}^{3}\sum_{m=0}^{3}G_{l,3}(u)G_{m,3}(v)\Delta P_{y(i+l,j+m)}$$

(*)

其中，$u=\frac{x}{N_x}-\left\lfloor\frac{x}{N_x}\right\rfloor$，$i=\left\lfloor\frac{x}{N_x}\right\rfloor-1$，$v=\frac{y}{N_y}-\left\lfloor\frac{y}{N_y}\right\rfloor$，$j=\left\lfloor\frac{y}{N_y}\right\rfloor-1$。

3.2 插值原理

3.2.1 插值基本思路

前面已经提到，无论是向前映射还是向后映射，一般情况下映射得到的点坐标不会刚好是整数，所以需要插值方法来得到映射后图像整数点的像素值。常见的插值方法有：最近领插值、双线性插值、双三次插值，本次作业我使用的也正是这三种。

本次作业我都是采用的向后映射，即遍历输出图像中的每一个坐标点，找到其对应的输入图像中的点坐标，再利用插值方法求得该点在输入图像中的像素值，令输出图像中对应坐标点的像素值等于该值。

3.2.2 最近邻插值

最近邻插值是最简单的一种插值方法。设输出图像中的坐标点(x_1, y_1)（整数点），在映射求得输入图像中点的坐标(x_0, y_0)（一般为非整数点）之后，对该点坐标四舍五入取整（实际是找到距离最近的坐标点），新得到的点(x_2, y_2)必然是整数点，(x_2, y_2)像素值也就是输出图像在对应点(x_1, y_1)的像素值。

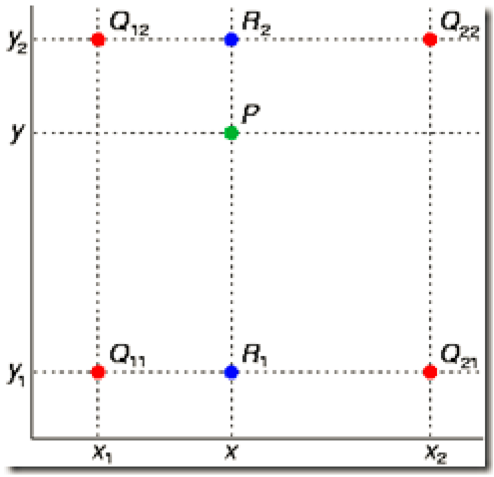
方法评价:

这种方法的优点是算法比较简单，运算速度很快。但是也存在严重的缺陷，例如边缘失真，产生锯齿。

3.2.3 双线性插值

设输出图像中整数点为(x_1, y_1)，在映射求得输入图像中点的坐标(x_0, y_0)之后，根据距离该点最近的四个坐标点求出该点像素值。

下面举个例子:



设 P 为插值点，$Q_{11}(x_1,y_1)$，$Q_{12}(x_1,y_2)$，$Q_{21}(x_2,y_1)$，$Q_{22}(x_2,y_2)$为 P 点周围四个网格点，需要根据这四个点确定 P 的像素值。

首先在 x 方向上插值，得到：

$$f(R_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}) \quad \text{Where} \quad R_1 = (x, y_1),$$
$$f(R_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}) \quad \text{Where} \quad R_2 = (x, y_2).$$

然后在 y 方向进行插值，得到：

$$f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2).$$

方法评价：

双线性插值算法比最近邻插值稍微复杂一些，但是比双三次插值更简单。其效果明显强于最近邻插值，但是比双三次插值略逊。

3.2.4 双三次插值

双线性的插值是三种插值方法中最复杂的。计算某一点的像素值需要利用其周围的 16 个点，所以最终得到的结果也会更加光滑。

其计算公式如下:

$$f(i+u,j+v)=\begin{pmatrix}S(u+1) & S(u) & S(u-1) & S(u-2)\end{pmatrix}f(i-1{:}i+2,j-1{:}j+2)\begin{pmatrix}S(v+1)\\ S(v)\\ S(v-1)\\ S(v-2)\end{pmatrix}$$

其中，$(i+u,j+v)$为映射到输入图像中的点，$u,v\in[0,1)$，

$S(x)$为插值基函数:

可以由以下式子近似:

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

方法评价：双三次插值的计算比较复杂，用到的点也比较多，其插值核函数和插值多项式都具有较好的连续性和可导性。因此其效果是三种插值方法中最好的。

4 方案设计

4.1 旋转扭曲方案

遍历输出图像中每一个像素点（x,y），根据前面说到底的旋转扭曲逆变换求出该点映射到输入图像中点(x_0,y_0)，然后根据选择的插值方法确定(x_0,y_0)点处像素值，将得到的像素值赋给（x,y）点。即完成扭曲变形。

4.2 水波纹变形方案

观察水波纹的映射公式可以发现，不论是正变换还是反变换都可以实现水波纹扭曲变形，所以不需要对前面提到的正变换公式做出改动。

具体方法与旋转扭曲相同，若要实现动态水波纹，则需要加入 Qt 中一个 Qtimer 类，每隔固定时间增加参数φ的值（我设定为每隔 150ms，φ增加 10）。

4.3 B 样条变形

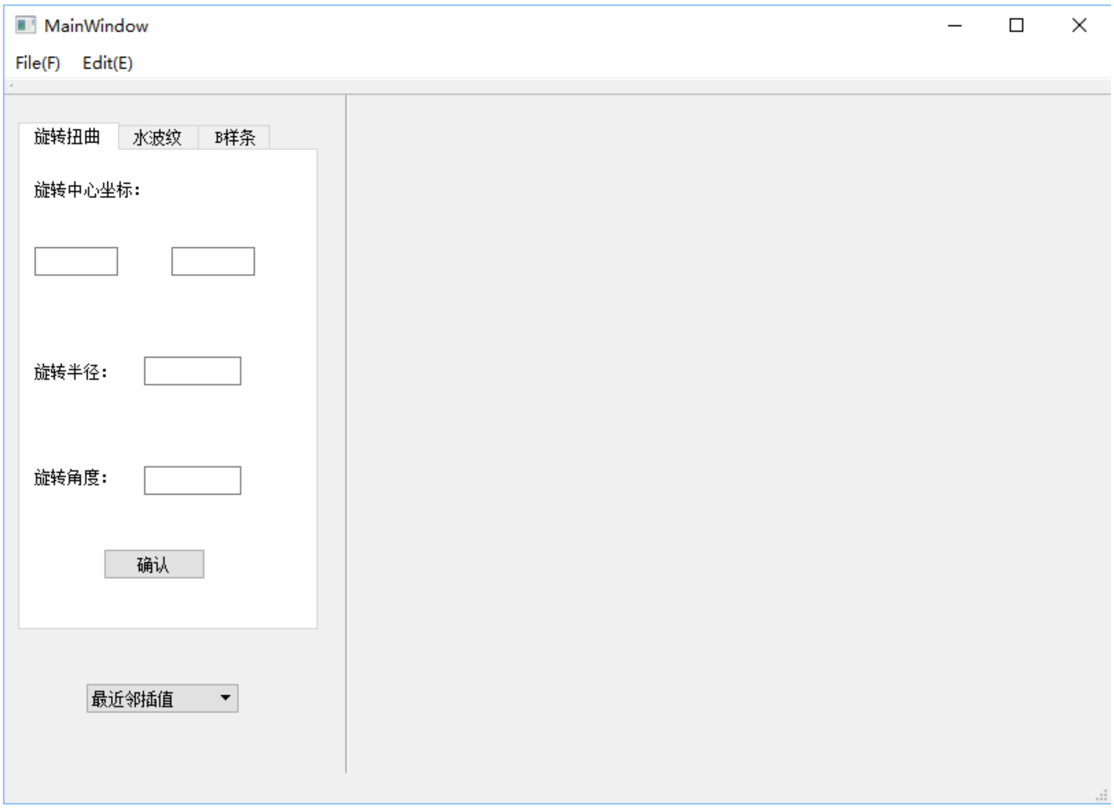
若强行求取 B 样条的反变换会涉及到求解高次方程，这将在很大程度上的限制变形速度。为了实现比较好的效果，且不影响变形速度，我的算法是：输出图像中的点减去根据公式（*）计算出的位移，便得到其对应的输入图像中点。

从原理上来说，这样的反变换肯定是不正确的，但是得到的最终效果还勉强达到预期。

本次作业每次 B 样条变换改变的控制点都只有一个，用户可以通过拖动实现点的移动，以拖动起始点为中心建立一个 4*4 的区域，每个区域的网格数 N_x*N_y 通过交互方式给定（N_x*N_y 越大，B 样条变形区域越大，变形越明显）。

5 结果分析

5.1 程序界面



本次作业程序界面如上，可以实现【文件打开】【文件保存】【一键还原】【撤销】等基本操作。

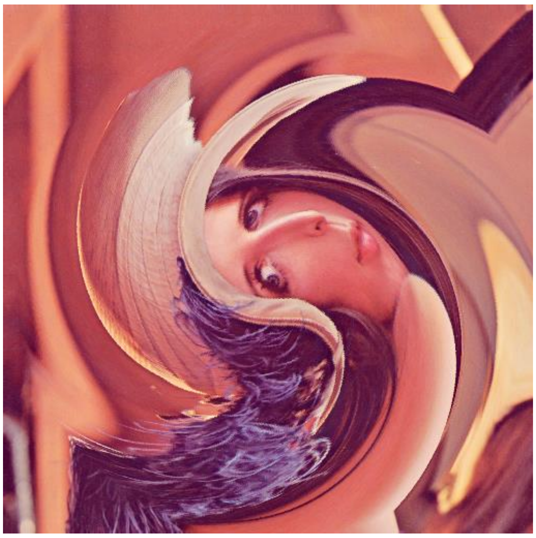
旋转扭曲：【旋转中心坐标】可以键盘输入，也可以鼠标点击图上的点。【旋转半径】和【旋转角度】都需要键盘输入，选择好插值方法之后点击【确定】即会显示变形结果。

水波纹：【中心点坐标】可以键盘输入，也可以鼠标点击。【水波纹参数设置】【半径】都需要键盘输入。若勾选了动态最终将会动态显示。选择好插值方法即可点击【确定】，此时便会显示水波纹变形结果。

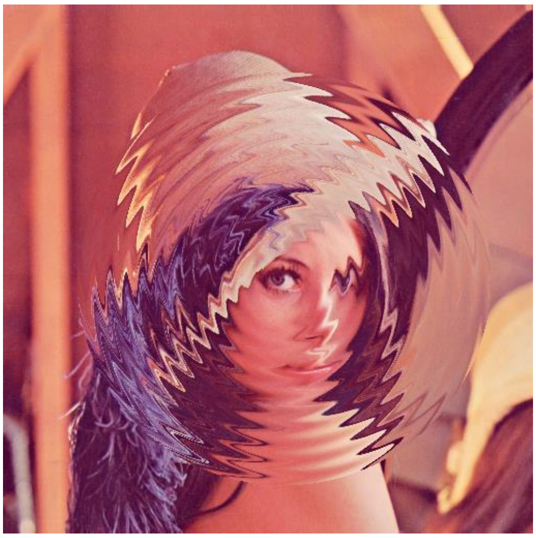
B 样条：键盘输入控制方格大小，点击确定，然后拖动图片上的点，拖动完成会立即显示 B 样条变形结果。

5.2 变形结果

旋转扭曲:



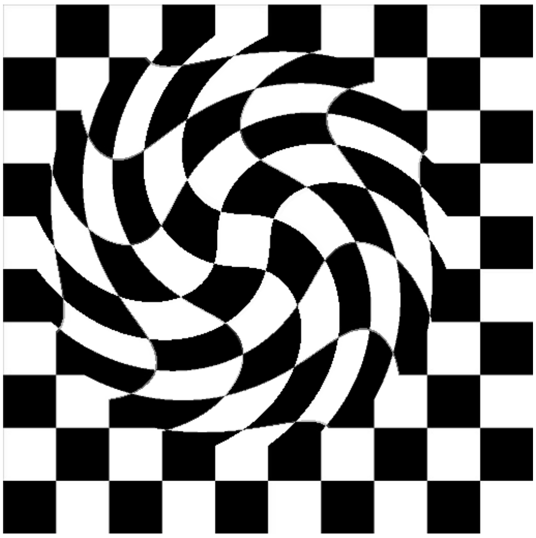
水波纹:



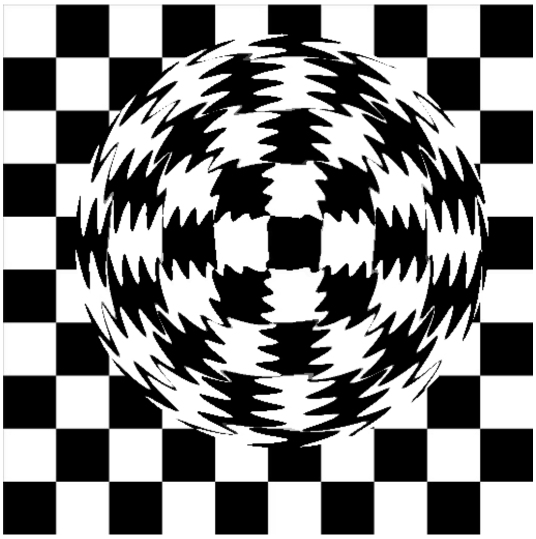
B 样条:

为了更加直观的显示变形效果，以下展示黑白格图片变形结果：

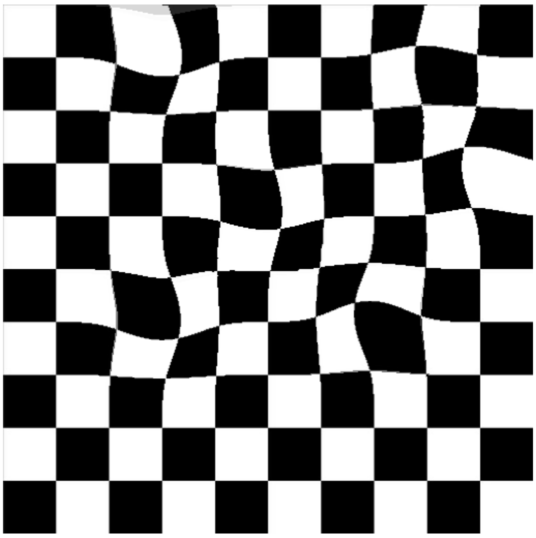
旋转扭曲：



水波纹：



B 样条:



以上均是采用最近邻插值的结果，可以明显的看到，最近邻插值的缺陷——很多地方出现了锯齿。

而双线性与双三次插值会好很多，下面比较一下不同插值方法的结果差异:

以旋转扭曲为例:

最近邻

双线性

双三次

观察上面的插值结果，明显可以看出双线性和双三次的优点。

6　误差分析

误差来源有模型误差、观测误差、方法误差和舍入误差四种。本次作业中无模型误差；观测误差来自读取原始图像 RGB 值时的误差，其上限为 0.5；考虑剩下的方法误差和舍入误差。

6.1　方法误差

方法误差视具体的插值方法确定。

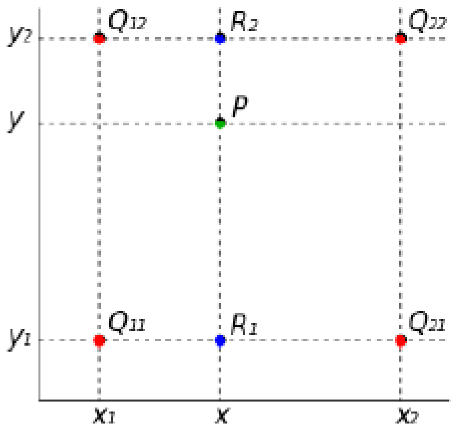
最近邻插值:

将(x,y)的值赋为距离其最近的(x^*,y^*)。像素值误差为:

$$\Delta|I| \le \left|\left(\frac{\partial f}{\partial x}\right)^*\right||\Delta x| + \left|\left(\frac{\partial f}{\partial y}\right)^*\right||\Delta y| \le 0.5\left(max\left|\frac{\partial f}{\partial x}\right| + max\left|\frac{\partial f}{\partial y}\right|\right)$$

双线性插值:

双线性的实质的分两次插值，一次是对 x 的插值，一次是对 y 的插值。以下面插值为例:



其中$Q_{11}(x_1,y_1)$，$Q_{12}(x_1,y_2)$，$Q_{21}(x_2,y_1)$，$Q_{22}(x_2,y_2)$，$R_1(x,y_1)$，$R_2(x,y_2)$，$P(x,y)$

固定 y 对 x 插值，此时的插值余项为:

$$|R_1(x,j)| = 0.5|\frac{\partial^2 f(\xi_1,j)}{\partial x^2}(x-i)(x-i-1)| \le 0.5M_1(\frac{i+1-i}{2})^2 = \frac{1}{8}M_1$$
$$|R_1(x,j+1)| = 0.5|\frac{\partial^2 f(\xi_2,j+1)}{\partial x^2}(x-i)(x-i-1)| \le 0.5M_1(\frac{i+1-i}{2})^2 = \frac{1}{8}M_1$$

其中，$M_1 = \max\limits_{i\le x\le i+1}\left|\frac{\partial^2 f(x,y)}{\partial x^2}\right|, j \le y \le j+1$。

同理，在 x 方向上的方法误差为: $\frac{1}{8}M_2$

其中，$M_2 = \max\limits_{j\le x\le j+1}\left|\frac{\partial^2 f(x,y)}{\partial y^2}\right|, i \le x \le i+1$

在计算 x 方向上的误差时，可以将 y 方向的插值误差看做观测误差。故有总误差为:

$$\frac{1}{8}(M_1+M_2)=\frac{1}{8}\left(\max_{i\le x\le i+1}\left|\frac{\partial^2 f(x,y)}{\partial x^2}\right|+\max_{j\le x\le j+1}\left|\frac{\partial^2 f(x,y)}{\partial y^2}\right|\right)$$

双三次插值:

首先证明双三次插值核函数的可导性。

S(x)在各个区间内部都是多项式，是任意阶连续可导的。考虑分段点-2，-1，0，1,2 即可。

先证明 S(x)的连续性，

$$S(0^+)=S(0^+)=1$$

$$S(1^+)=0$$

$$S(1^-)=0$$

$$S(2^+)=0$$

$$S(2^-)=0$$

以上说明，在 0,1,2 点 S(x)连续，由对称性可得-1，-2 点依旧连续。

证明$S'(x)$的连续性，

$$S'(\mathrm{x})=\begin{cases}8+10\mathrm{x}+3x^2, & -2<\mathrm{x}<-1\\ -4\mathrm{x}-3x^2, & -1<\mathrm{x}<0\\ -4x+3x^2, & 0<x<1\\ -8+10x-3x^2, & 1<x<2\\ 0, & otherwise\end{cases}$$

$$S'(-2^-)=0$$

$$S'(-2^+)=0$$

$$S'(-1^-)=1$$

$$S'(-1^+) = 1$$

$$S'(0^-) = 0$$

$$S'(0^+) = 0$$

$$S'(1^-) = -1$$

$$S'(1^+) = -1$$

$$S'(2^-) = 0$$

$$S'(2^+) = 0$$

所以，$S'(\mathrm{x})$是连续的。

通过双三次插值得到的插值函数是一阶偏导连续的，这个结果类似于埃尔米特插值。所以这里我直接借用埃尔米特插值的余项公式，得到误差：

$$\begin{cases} |R_3(x)| = \left|\dfrac{1}{4!} {f_x}^{(4)}(\xi){\omega_2}^2(x)\right| \le \dfrac{1}{384} M_{4x}{h_x}^2 \\ |R_3(y)| = \left|\dfrac{1}{4!} {f_y}^{(4)}(\xi){\omega_2}^2(y)\right| \le \dfrac{1}{384} M_{4y}{h_y}^2 \end{cases}$$

6.2 舍入误差

运算过程中我都是采用的 float 类型，其舍入误差为$\frac{1}{2} \times 10^{-5}$

计算得到 RGB 的值需要转换为整数型，这里存在的误差上限为 0.5。

考虑到两者数量级差别比较大，所以最终舍入误差应该取为 0.5。

7 参考文献、网站

[1] 李庆扬，王能超，易大义 数值分析 清华大学出版社，第5版，2008
[2] 维基百科“B样条”：https://zh.wikipedia.org/wiki/B样条