作业 02 Graph Warping

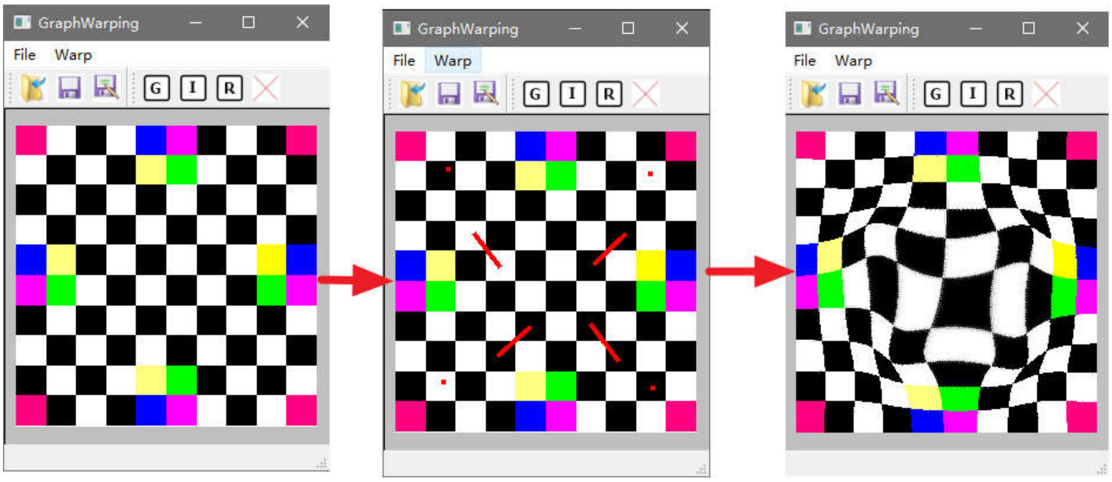
庄涛 PB15111679,, ID：85

计算机科学与技术系，215 院 011 系 01 班

2018/3/11

一、内容

写一个图像扭曲程序 Graph Warping，要求实现算法 IDW 和 RBF。



1.1 IDW 算法

$$f(p)=\sum_{i=1}^{n} w_i(p) f_i(p)$$

where $f(p)=y_i, i=1,......,n. w_i : R^2 \to R$ is the weight function, which must satisfy the condition:

$$w_i(p_i)=1, \sum_{i=1}^{n} w_i(p)=1, \text{ and } w_i(p)\geq 0, i=1,........,n.$$

These conditions guarantee the property of interpolation. Shepard proposed the following simple weight function:

$$w_i(p)=\frac{\sigma_i(p)}{\sum_{j=1}^{n}\sigma_j(p)} \text{ with } \sigma_j=\frac{1}{d(p,p_i)^{\mu}}$$

Where $\mathrm{d}(\mathrm{p},\mathrm{p}_i)$ is the distance between p and p_i.

1.2 RBF 算法

$$T(x) = \sum_{i=1}^{N} a_i g(\|x - x_i\|) + Aq + \alpha_3$$

where $A = (\alpha_1, \alpha_2)^T, \alpha_i \in R^2, 1 \le i \le N, \alpha_i = \{\alpha_i^x, \alpha_i^y\} \in R^2$. Here $\|\cdot\|$ denotes the usual

Euclidean norm on R^2 and $g: R^+ \to R$. $T(x_i) = F_i, 1 \le i \le N$.

The system of equations for the vectors of unknowns is:
$u_k = (a_1^k, ..., a_N^k)^T$ and $v_k = (\alpha_1^k, \alpha_2^k, \alpha_3^k)^T$, $k = 1,2$. The linear system is described as:

$$Gu_k + Hv_k = b_k$$
$$H^T u_k = 0$$

Here $b_k = (y_1^k, ..., y_N^k)$, $G = \{g(\|q_i - q_j\|)\}_{i,j=1}^{N}$, and H is an $N \times 3$ matrix with ith row

$\{q_i^1, q_i^2, 1\}, 1 \le i \le N$.

- ***Choose the basis function***

  Well-known radial basis functions are multi-quadrics, originally proposed by R.Hardy:

  $R(d) = (d^2 + r^2)^{\mu/2}$ with $r > 0$ and $\mu \neq 0$

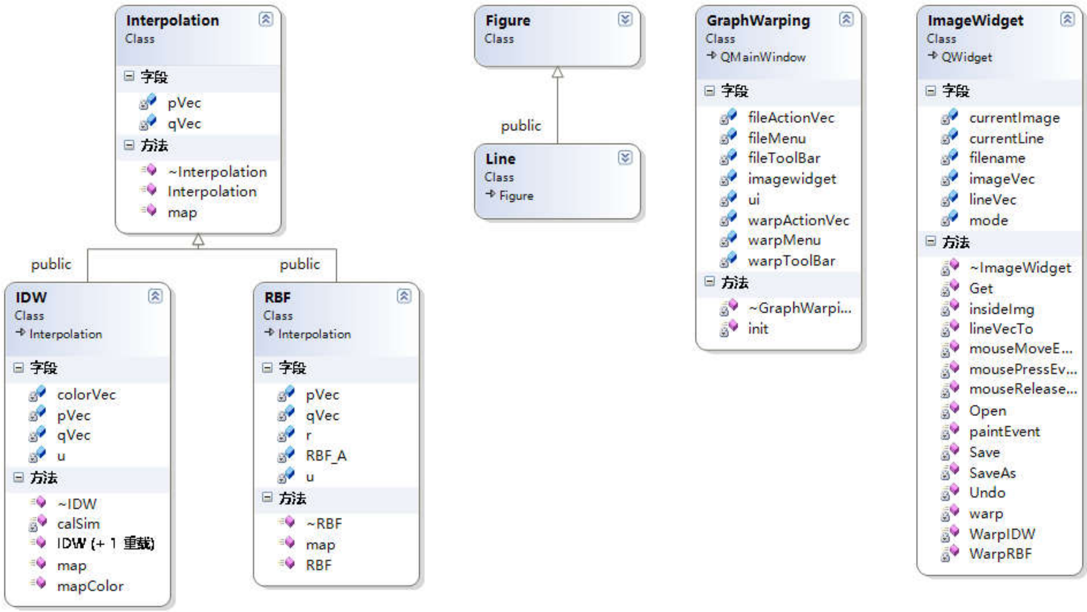
  Here, $\mu = \pm 1$ has been used successfully.

二、环境

System：Windows 10
IDE：Microsoft Vistual Studio 2010
依赖库：QT-5.5，eigen-3.3.2

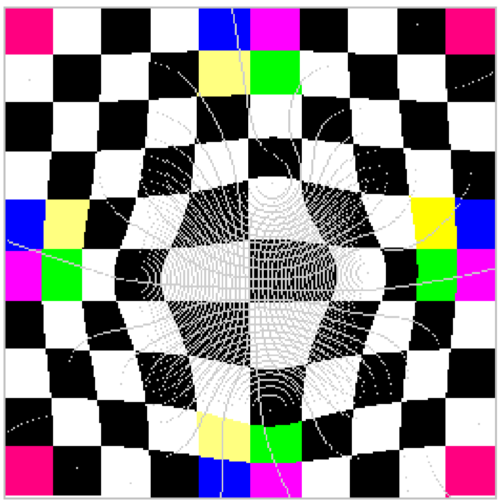
三、过程

3.1 类关系图



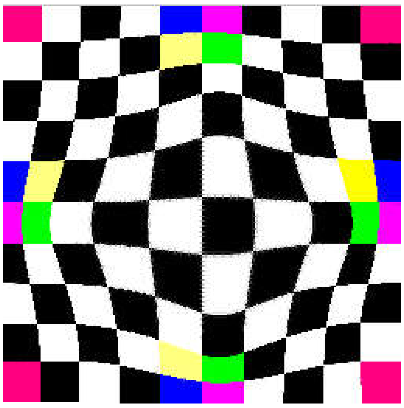
3.2 IDW

Idw 方法的核心就是在原位置的基础上对各点的偏移加权求和，权值依赖于两点的距离。实现较为简单。但效果存在一定的问题，如下



图像在弯曲变换过程中，会产生白纹。原因在于图像上离散的，没法与新图像各像素一一映射。为了解决这个问题，可以考虑采用小范围插值的方法，即用空白点周围的像素的颜色来获得空白点的颜色。这相当于在二维空间上的三维（rgb）插值问题。只需稍微在原来的 idw 方法上稍微修改即可。

修改后的效果如下



3.3 RBF

RBF 的核心也是在原位置上对与距离有关的量加权求和，权值只依赖于插值点。实现的难点就在于各系数的计算。线性方程组如下

$$Gu_k + Hv_k = b_k$$
$$H^T u_k = 0$$

转化为 Ax=b 的形式

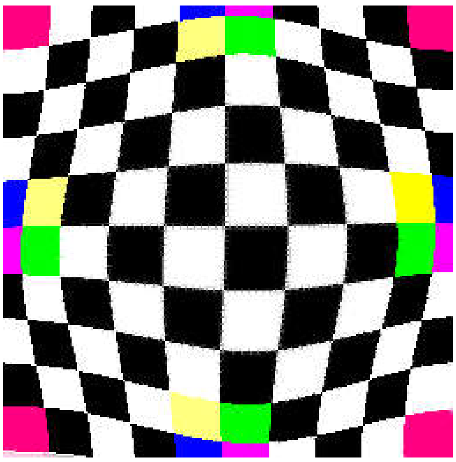
$A = \begin{bmatrix} G & H \\ H^T & 0 \end{bmatrix}$， $b = \begin{bmatrix} p_x \\ 0 \end{bmatrix}$， 解得 $x = \begin{bmatrix} a_x \\ \alpha_x \end{bmatrix}$， （N+3） x 1

以及

$A = \begin{bmatrix} G & H \\ H^T & 0 \end{bmatrix}$， $b = \begin{bmatrix} p_y \\ 0 \end{bmatrix}$， 解得 $y = \begin{bmatrix} a_y \\ \alpha_y \end{bmatrix}$， （N+3） x 1

因此有$RBF_A = \begin{bmatrix} x^T \\ y^T \end{bmatrix}$， 2 x （N+3），所得插值函数即为$f(p) = RBF_A * \begin{bmatrix} d \\ p \\ 1 \end{bmatrix}$， 其中 RBF_A 是全系数矩阵， d 是$(g(\|p-p_1\|), g(\|p-p_2\|), \dots, g(\|p-p_N\|)\ )^T$。

效果示范如下



3.4 Undo

为了图像编辑方便，特地添加了 undo 功能，语意视情况而定。当在获取数据时，undo 可以撤销所画的线；当不在获取数据时，undo 可以恢复成上一个图像。

四、复杂度分析

假设图像的宽度为 w，高度为 h，插值点个数为 n

4.1 idw

单个像素的插值计算复杂度为$\theta(n)$，因此 idw 的复杂度为$\theta(whn)$

4.2 rbf

求解的 Ax=b 方程中 A 的维度为（n+3）x（n+3），因此方程求解的复杂度为$O(n^2)$，故单个像素的插值计算复杂度为$O(n^2)$，因此总的时间复杂度为$O(whn^2)$

4.3 填补空白

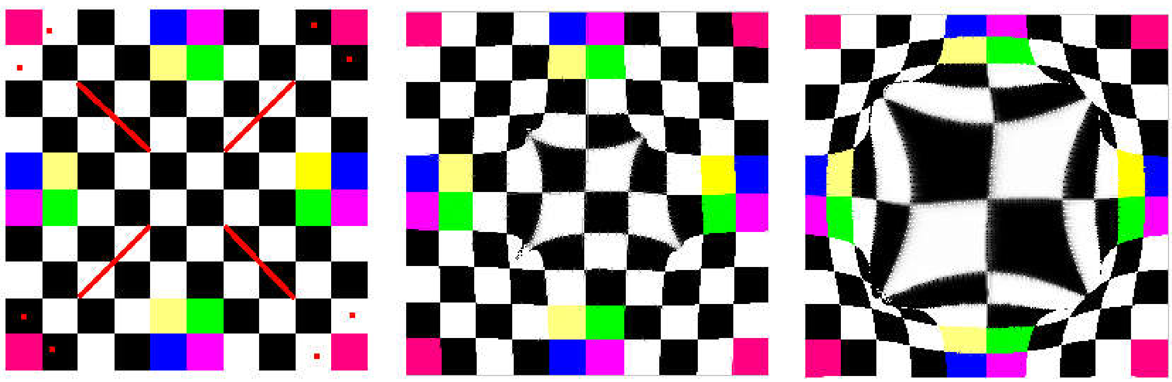
假设空白的点数为 m，小范围正方形的宽度为 k

则每个空白点插值的复杂度为$\theta(whnk^2)$

因此将所有空白点补上的时间复杂度为$\theta(mwhnk^2)$

五、结果分析

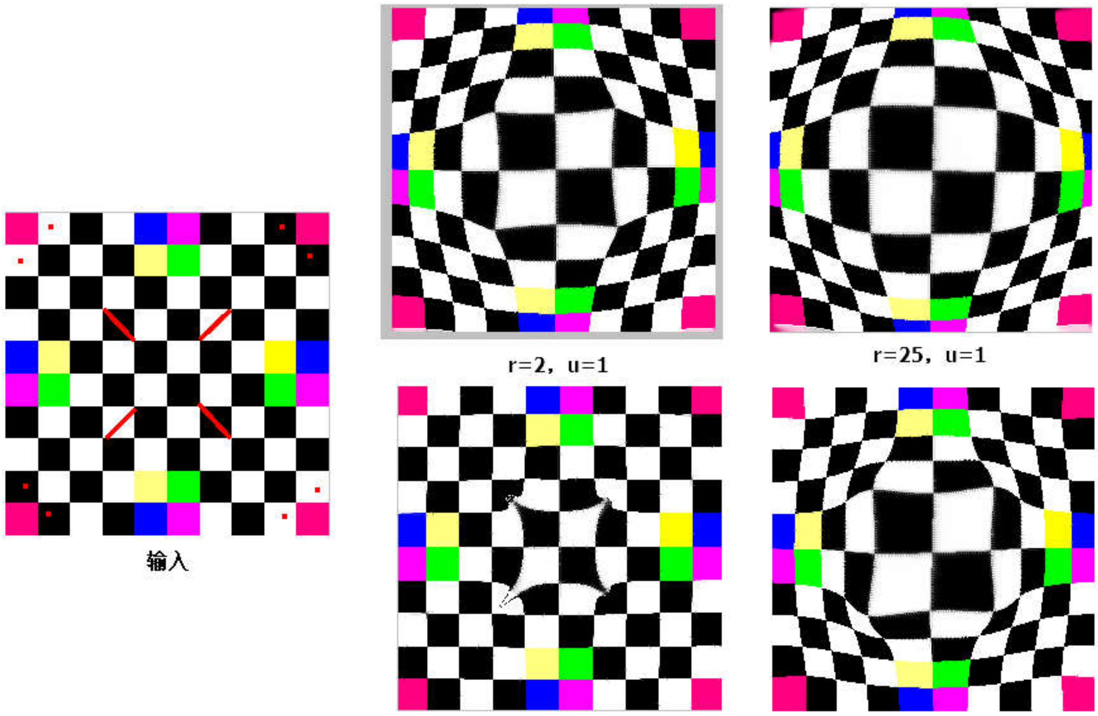
5.1 idw



输入　　u = 0.5　　u = 2

u 越大，距离越远影响越小，因此某一像素点更多的受到相邻的数据点的影响。可以见到，当 u=0.5 时，中间部分的点的变化量较小，原因在于各个数据点的影响都比较显著，因此比较平衡。当 u=2 时，中间部分的点的变化量较大，原因在于这些点更多的受到其相邻数据点的影响，因此比较失衡。

5.2 rbf



输入

r=2，u=1　r=25，u=1

r=2，u=-1　r=25，u=-1

规律大概是 r 越大，u=1 就是影响范围越大。

六、总结

开始涉及了二维图像的相关实践，加深了对插值运算的理解，继续锻炼了 C++的编程能力。