# 《数字图像处理》作业 2

学号：SA21229075　　姓名: 陈文博

2020 年 10 月 05 日

## 1　作业要求：

实现如下图像变形算法

- **IDW warping:**: Inverse distance-weighted interpolation method Detlef Ruprecht and Heinrich Müller. Image warping with scattered data interpolation. IEEE Computer Graphics and Applications, 1995.
- **RBF warping:**: Radial basis function interpolation method Nur Arad and Daniel Reisfeld. Image Warping Using Few Anchor Points and Radial Functions. Computer Graphics Forum, 14(1): 35-46, 1995.

## 2　开发环境

**IDE**: Microsoft Visual Studio 2019 Community

**CMake**: 3.14.0

**相关依赖库**: std_image, spdlog, imgui, glm, glfw, glad, entt, OpenCV, OpenMP 等

## 3 交互界面

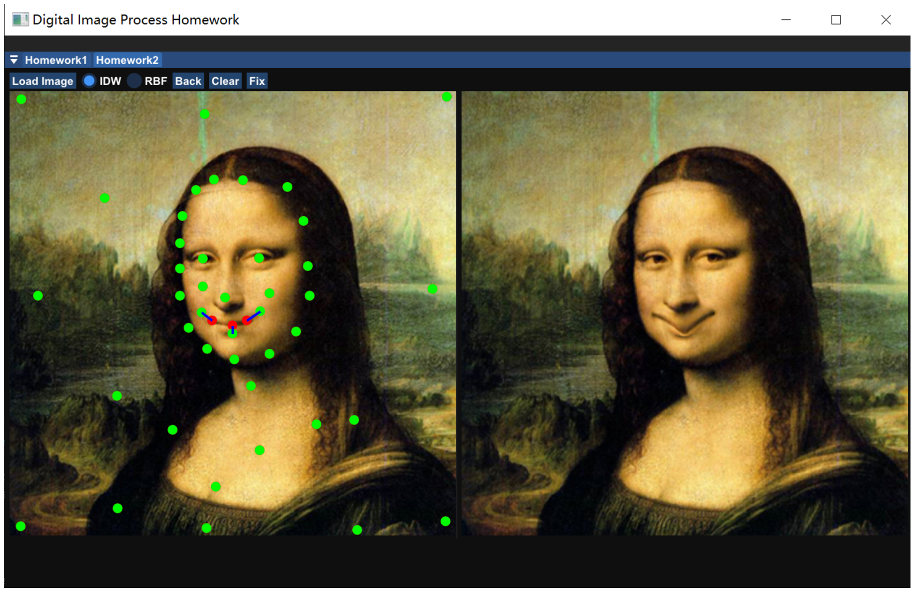


图 1: 交互界面说明

- 使用 OpenMP 进行并行化加速，充分发挥多核处理器性能
- 在不进行填充修复的情况下能够实现实时拖拽编辑，显示算法结果

## 4 算法原理

### 4.1 基本原理

- 输入：$n$ 对控制点对 $(\boldsymbol{p}_i, \boldsymbol{q}_i)$，$i = 1, 2, \cdots, n$，其中 $\boldsymbol{p}_i \in \mathbb{R}^2$ 为控制起始点，$\boldsymbol{q}_i \in \mathbb{R}^2$ 为控制目标点
- 目标：找到一个映射 $f : \mathbb{R}^2 \to \mathbb{R}^2$，满足 $f(\boldsymbol{p}_i) = \boldsymbol{q}_i$，$i = 1, 2, \cdots, n$

## 4.2 Inverse distance-weighted interpolation methods(IDW) [2]

IDW 算法基本原理是根据给定的控制点对和控制点对的位移矢量，计算控制点对周围像素的反距离加权权重影响，实现图像每一个像素点的位移。

选择 $n$ 对控制点对 $(\boldsymbol{p}_i,\boldsymbol{q}_i)$，$i=1,2,\cdots,n$，目标映射 $f:\mathbb{R}^2\to\mathbb{R}^2$ 可表示成以下形式：

$$f(\boldsymbol{p})=\sum_{i=1}^{n}\omega_i(\boldsymbol{p})f_i(\boldsymbol{p}) \tag{1}$$

其中，权重 $\omega_i(\boldsymbol{p})$ 满足：

$$w_i(\boldsymbol{p})=\frac{\sigma_i(\boldsymbol{p})}{\sum_{j=1}^{n}\sigma_j(\boldsymbol{p})} \tag{2}$$

$\sigma_i(\boldsymbol{p})$ 反映第 $i$ 对控制点对像素 $\boldsymbol{p}$ 得反距离加权权重影响程度，可以直接取：

$$\sigma_i(\boldsymbol{p})=\frac{1}{\|\boldsymbol{p}-\boldsymbol{p}_i\|^{\mu}} \tag{3}$$

其中 $\mu>1$，也可以取 locally bounded weight：

$$\sigma_i(\boldsymbol{p})=\left[\frac{R_i-d(\boldsymbol{p},\boldsymbol{p}_i)}{R_i d(\boldsymbol{p},\boldsymbol{p}_i)}\right]^{\mu} \tag{4}$$

$f_i$ 为线性函数，满足：

$$f_i(\boldsymbol{p})=\boldsymbol{q}_i+\boldsymbol{T}_i(\boldsymbol{p}-\boldsymbol{p}_i) \tag{5}$$

其中 $\boldsymbol{T}_i$ 为二阶矩阵：

$$\boldsymbol{T}_i=\begin{bmatrix} t_{11}^{(i)} & t_{12}^{(i)} \\ t_{21}^{(i)} & t_{22}^{(i)} \end{bmatrix} \tag{6}$$

矩阵 $\boldsymbol{T}$ 得确定，可以通过求解如下最优化问题：

$$\arg\min_{\boldsymbol{T}_i} E(\boldsymbol{T}_i)=\sum_{j=1,j\neq i}^{n}\sigma_i(\boldsymbol{p}_j)\|\boldsymbol{q}_j-f_i(\boldsymbol{p}_j)\|^2 \tag{7}$$

上式对 $\boldsymbol{T}_i$ 求导，令方程为 0 得：

$$\sum_{j=1,j\neq i}^{n}\sigma_i(\boldsymbol{p}_j)\left(\boldsymbol{p}^T\boldsymbol{q}-\boldsymbol{p}^T\boldsymbol{T}_i\boldsymbol{p}\right)=0 \tag{8}$$

其中 $\boldsymbol{p}=\boldsymbol{p}_j-\boldsymbol{p}_i$，$\boldsymbol{q}=\boldsymbol{q}_j-\boldsymbol{q}_i$，通过变形可得到：

$$\boldsymbol{T}_i\sum_{j=1,j\neq i}\sigma_i(\boldsymbol{p}_j)\boldsymbol{p}\boldsymbol{p}^T=\sum_{j=1,j\neq i}\sigma_i(\boldsymbol{p}_j)\boldsymbol{q}\boldsymbol{p}^T \tag{9}$$

又 $\sigma_i(\boldsymbol{p}_j)\boldsymbol{p}\boldsymbol{p}^T$ 非奇异，因此可以直接解出 $\boldsymbol{T}_i$ 的值：

$$\boldsymbol{T}_i=\left(\sum_{j=1,j\neq i}\sigma_i(\boldsymbol{p}_j)\boldsymbol{q}\boldsymbol{p}^T\right)\left(\sum_{j=1,j\neq i}\sigma_i(\boldsymbol{p}_j)\boldsymbol{p}\boldsymbol{p}^T\right)^{-1} \tag{10}$$

求出 $\boldsymbol{T}_i(i=1,\cdots,n)$ 后，映射 $f$ 也就相应确定

### 4.3 Radial basis functions interpolation method(RBF) [1]

选择 $n$ 对控制点对 $(\boldsymbol{p}_i,\boldsymbol{q}_i)$，$i=1,2,\cdots,n$，目标映射 $f:\mathbb{R}^2\to\mathbb{R}^2$ 可表示为以下形式：

$$f(\boldsymbol{p})=\sum_{i=1}^{n}\alpha_i g_i(\|\boldsymbol{p}-\boldsymbol{p}_i\|)+\boldsymbol{A}\boldsymbol{p}+\boldsymbol{B} \tag{11}$$

其中，$g_i$ 为径向基函数，通常可以取 Hardy multiquadrics：$g(t)=(t^2+c^2)^{\pm\frac{1}{2}}$ 或高斯函数 $g_\sigma(t)=e^{-t^2/\sigma^2}$，为了计算方便，这里取 Hardy multiquadrics：

$$\begin{aligned} g_i(d)&=(d+r_i)^{\pm\frac{1}{2}} \\ r_i&=\min_{j\neq i}d(\boldsymbol{p}_i,\boldsymbol{p}_j) \end{aligned} \tag{12}$$

对于线性部分分量 $\boldsymbol{A}\boldsymbol{p}+\boldsymbol{B}$，本例简单地取 $\boldsymbol{A}=\boldsymbol{I}$ 和 $\boldsymbol{B}=\boldsymbol{0}$

# 5 实验效果

## 5.1 标准图像测试

如下图所示，固定四角，蓝色为控制起始点，绿色为控制终止点

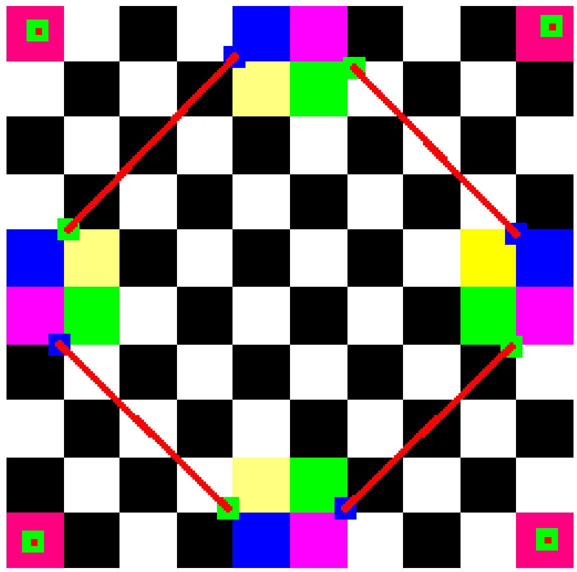

图 2: 拉伸情况

## 5.2 IDW 算法

### 5.2.1 $\mu = -1$ 情况

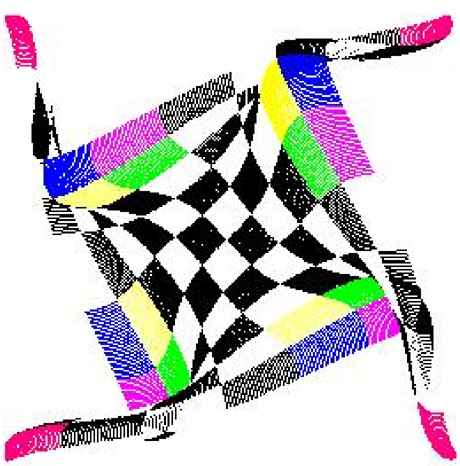

图 3: 修复前

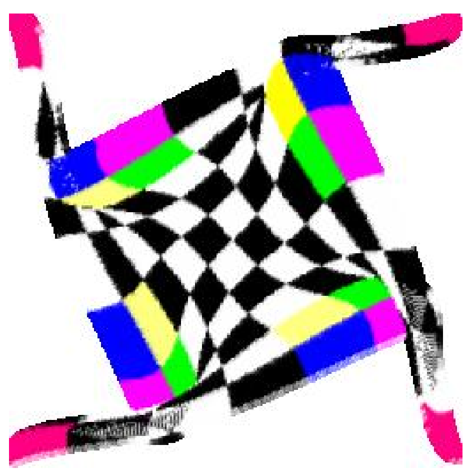

图 4: 修复后

#### 5.2.2 $\mu = 1$ 情况

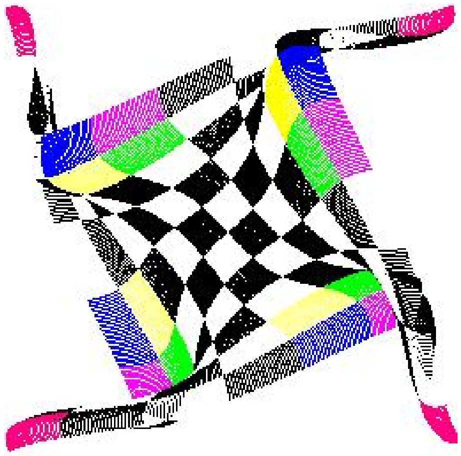

图 5: 修复前

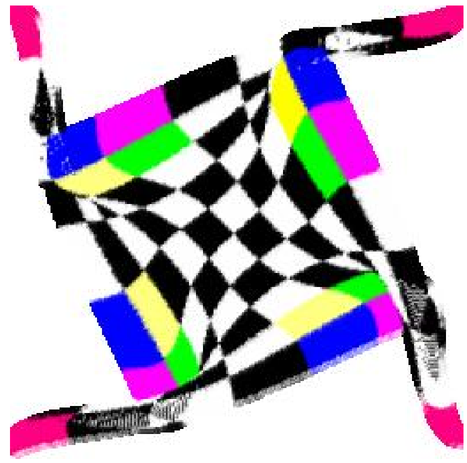

图 6: 修复后

### 5.3 RBF 算法

#### 5.3.1 $\mu = 0.5$ 情况

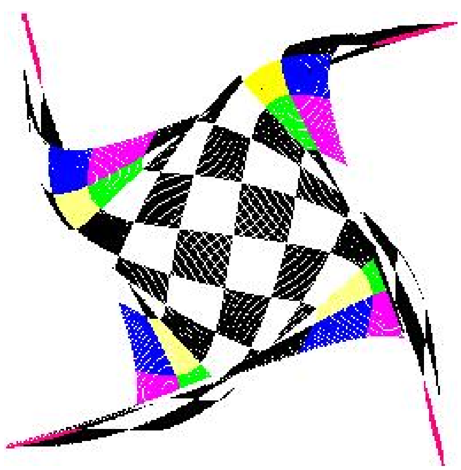

图 7: 修复前

图 8: 修复后

#### 5.3.2 $\mu = -0.5$ 情况

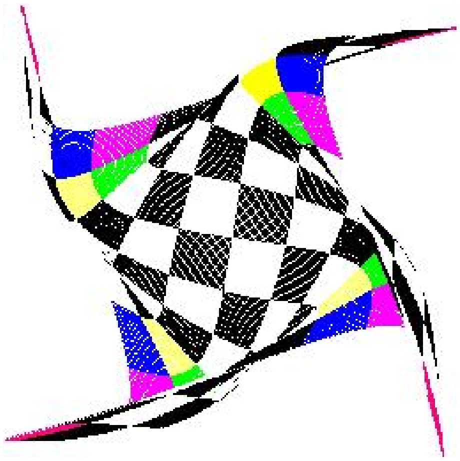

图 9: 修复前

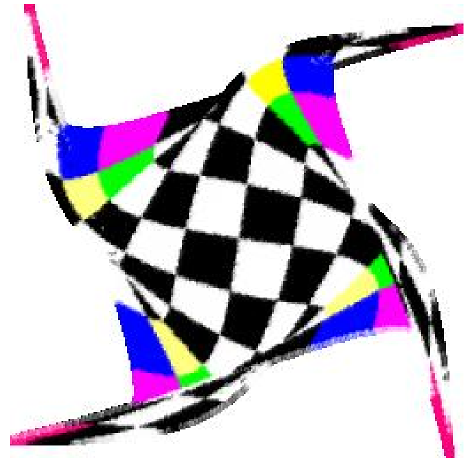

图 10: 修复后

## 5.4 其他测试

### 5.4.1 柴犬表情包

**原图片:**

图 11: 原始图片

**处理后：**

图 12: Happy

图 13: Emmm...

### 5.4.2 面带表情的藏狐


图 14: 未处理


图 15: 处理后

## 6 总结

本例中使用 IDW 和 RBF 两种方法进行图像的拉伸变换，理论上 IDW 和 RBF 的运算复杂度均为 $O(n^2+nN)$，而由于实际运算中，IDW 计算一个像素点的浮点乘法加法次数比 RBF 方法更多，在实验中也可以发现 RBF 处理速度要比 IDW 快 3 到 4 倍

# A 附录

实验源码地址：https://github.com/Chaphlagical/DIP

项目构建、编译方法于 README.md 中给出

# 参考文献


[1] N. Arad and D. Reisfeld. Image warping using few anchor points and radial functions. In *Computer graphics forum*, volume 14, pages 35–46. Wiley Online Library, 1995.

[2] D. Ruprecht and H. Muller. Image warping with scattered data interpolation. *IEEE Computer Graphics and Applications*, 15(2):37–43, 1995.