人脸变形——数值大作业一

林嘉成 2016011498
(自动化系 自 66)

目录

1 需求分析

人脸变形即，在引导图的面部 68 个关键点的引导下，将源图的面容进行扭曲变形，使得得到的图片的人脸的关键点特征与导引图的关键点特征相似。

由于该变形过程无法用显式的数学公式进行表达，在变形上存在一定的难度。而我们的**需求**是将源图的关键点坐标的位置映射到导引图的关键点坐标的位置，同时也要将关键点坐标附近的像素坐标也映射过去。由于映射之后得到的坐标点不是整数，所以需要进行插值。为此，可以采用两个常见的变形函数，即 B 样条变形和 TPS 变形。

在本报告中，规定源图为待变形图，导引图为将源图的关键点特征变换成的目标的图片，生成图即变形后的图片。如将特朗普的脸的特征换成梁静茹的儿子 Anderson 的脸，源图为特朗普，导引图为 Anderson。

2 TPS 变形

2.1 方案设计

首先将导引图的关键点经平移、放大、旋转等操作进行仿射变换以变换到源图的关键点的位置，进行粗对齐。之后以导引图的关键点为控制点，源图的关键点为目标点，进行 TPS 求解即得到变形之后的图像。

2.2 方案基本原理

2.2.1 仿射变换参数拟合

仿射变换是最常用的空间坐标变换之一，其形式如下。

$$[x, y, 1] \begin{bmatrix} a & d \\ b & e \\ c & f \end{bmatrix} = [x', y']$$

通过仿射变换拟合参数之后，坐标点进行了尺度、旋转、平移或偏移。对参数进行最小二乘法拟合，在进行仿射变换之后，将源图和导引图的人脸进行粗对齐，便于后续 TPS 操作。

2.2.2 TPS 变形

TPS(Thin plate spline) 是一种常见的插值模型，目标是寻找一个通过所有控制点的光滑曲面 $f(x, y)$，使得能量函数 I_f 最小, 且该问题有解析解。

$$I_f = \iint_{R^2} \left(\left(\frac{\partial^2 f}{\partial x^2} \right)^2 + 2 \left(\frac{\partial^2 f}{\partial x \partial y} \right)^2 + \left(\frac{\partial^2 f}{\partial y^2} \right)^2 \right) dxdy$$

TPS 求解 给定 n 个控制点 $P_1 = (x_1, y_1), ..., P_n = (x_n, y_n)$，记矩阵 K,P,L 分别为

$$K = \begin{bmatrix} 0 & U(r_{12}) & ... & U(r_{1n}) \\ U(r_{21}) & 0 & ... & U(r_{2n}) \\ ... & ... & ... & ... \\ U(r_{n1}) & U(r_{n2}) & ... & 0 \end{bmatrix} P = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ ... & ... & ... \\ 1 & x_n & y_n \end{bmatrix} L = \begin{bmatrix} K & P \\ P^T & \mathbf{0} \end{bmatrix}$$

假定目标点为 $P_1^{'} = (x_1^{'}, y_1^{'}), ..., P_n^{'} = (x_n^{'}, y_n^{'})$，记矩阵 V,Y 为

$$V = \begin{bmatrix} x_1^{'} & x_2^{'} & ... & x_n^{'} \\ y_1^{'} & y_2^{'} & ... & y_n^{'} \end{bmatrix} Y = \begin{bmatrix} V & \mathbf{0} \end{bmatrix}^T$$

其中 $U(r)$ 为径向基函数，即 $U(r) = r^2 \log(r^2)$ $r \neq 0$,r 为坐标点之间的距离。任意一点的坐标为

$$f(x,y) = [f_x(x,y), f_y(x,y)]^T = a_1 + a_x x + a_y y + \sum_{i=1}^{n} \omega_i U(|P_i - (x,y)|)$$

其中，a_1, a_x, a_y, ω 为线性方程组 $L\,[\omega_1, ..., \omega_n, a_1, a_x, a_y]^T = Y$ 的解。

2.2.3 具体实现

控制点为导引图的 68 个关键点，目标点为源图的 68 个关键点，即 P_i 为导引图的关键点，$P_i^{'}$ 为源图的关键点，注意导引图的关键点要带入仿射变换之后的坐标。带入方程计算求解再进行插值即得到变形后的图片。

Algorithm 1 TPS 变形

Require: 导引图的 68 个关键点 P_i，源图的 68 个关键点 $P_i^{'}$
Ensure: 变形后的图像

1: **function** TPS 变换 (x', y', x, y)
2: 做 (x', y') 到 (x, y) 的仿射变换实现特征点对齐
3: 构造 L,Y 矩阵
4: 解线性方程组 $L\,[\omega_1, ..., \omega_n, a_1, a_x, a_y]^T = Y$
5: 将参数代入 $f(x,y) = [f_x(x,y), f_y(x,y)]^T = a_1 + a_x x + a_y y + \sum_{i=1}^{n} \omega_i U(|P_i - (x,y)|)$
6: ImgDst.SetPixel 并进行插值。
7: **return** ImgDst

3 B 样条变形

3.1 方案设计

首先将导引图的关键点经平移、放大、旋转等操作进行仿射变换以变换到源图的关键点位置以进行粗对齐。之后通过使用三次 B 样条扭曲进行人脸变形，即得到变换之后的结果。其中，网格点的移动受控制点影响，姑且将权重全部设为 1，之后再使用梯度下降方法以求得每一个网格点的位移权重，得到使得 Loss 较小的权重，用该组参数进行变形即得到目标图片。

3.2 方案基本原理

3.2.1 B 样条变形

给定 m+n+1 个平面或空间 $P_i(i = 0, 1, ..., m+n)$，称 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 $k = 0, 1, 2, ..., m$，这些曲线段的全体称为 n 次 B 样条曲线，其顶点 $P_i(i = 0, 1, ..., n+m)$ 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。本项目主要使用的是三次 B 样条曲线的基函数。

三次 B 样条曲线基函数

$$G_{0,3}(t) = \frac{1}{6}(-t^3 + 3t^2 - 3t + 1)$$
$$G_{1,3}(t) = \frac{1}{6}(3t^3 - 6t^2 + 4)$$
$$G_{2,3}(t) = \frac{1}{6}(-3t^3 + 3t^2 + 3t + 1)$$
$$G_{3,3}(t) = \frac{1}{6}t^3$$

其中，$t \in [0, 1]$.

三次 B 样条扭曲 B 样条扭曲由网格点的位移影响其附近图像的变形。三次 B 样条扭曲中，任一点的位移由距其最近的 16 个网格点的位移决定。设图中某一点的的坐标为 (x, y)，设网格大小为 N，则有

$$x_i = \lfloor \frac{x}{N} \rfloor \quad x_u = \frac{x}{N} - \lfloor \frac{x}{N} \rfloor$$
$$y_i = \lfloor \frac{y}{N} \rfloor \quad y_u = \frac{y}{N} - \lfloor \frac{y}{N} \rfloor$$

其中，坐标 (x_i, y_i) 为网格点 P_{x_i,x_j} 的坐标。根据这一原则将图片建立起网格点坐标系，则可以得到点 (x, y) 的位移为

$$s(x, y) = \sum_{k=0}^{3}\sum_{l=0}^{3} G_{k,3}(x_u)G_{l,3}(y_u)s(P_{i+k-1,j+l-1})$$

3.2.2 梯度下降法

梯度下降的主要思想为：开始时随机选取一个参数的组合，即 $(\theta_1, \theta_2, ..., \theta_n)$，计算代价函数，然后寻找下一个能让代价函数值下降最多的参数组合，不断迭代后得到一个局部最小值。

这里为了得到较好的网格点位移权重，故采用梯度下降法。定义 Loss 函数为

$$J(\theta_1, \theta_2, ..., \theta_n) = \frac{1}{2m}\sum_{i=1}^{m}\left(s(P_i) - \sum_{k=0}^{3}\sum_{l=0}^{3} G_{k,3}(x_{i_u})G_{l,3}(y_{i_u})s(P_{i+k-1,j+l-1})\theta_{i+k-1,j+l-1}\right)^2$$

其中，m 为脸部关键点个数 (一般取 68)，n 为网格点个数。s(•) 表示的是点的位移，θ 表示的是网格点位移权重。设学习率为 α，每次迭代进行参数的同步更新，即

$$\theta_i := \theta_i - \alpha\frac{\partial J}{\partial \theta_i}$$

当迭代到一定次数或 Loss 函数的值小于某个阈值的时候梯度下降停止。

要注意学习率 α 的选取：如果 α 太大，则梯度下降法可能会越过最低点，甚至可能无法收敛，即移动步数很大导致每次更新都越过最低点，离最低点越来越远；如果 α 太小，梯度下降可能会很慢。这里默认学习率为 0.001，且为了避免无法收敛的情况，在具有发散的趋势时，将当前学习率减半，以保证最终代价函数能够收敛。

3.2.3 具体实现

先定义三类点，即控制点、网格点、任一点。控制点即为脸部的 68 个关键点，网格点定义见上文，任一点即变换后的图像上的每一个点。现在所需要做的，首先建立好网格坐标，之后根据控制点的位移计算每个控制点周围的 16 个点的位移偏差。即遍历 68 个关键点，周围的每个网格点以 1 的权重进行分别求和。由于权重为 1，故每个网格点的位移的权重需要进行调整，故对所有网格点设一个位移权重，采用梯度下降法进行求解。设 X,Y 方向的学习率均为 0.001，初始化权重均为 0.05。

Algorithm 2 三次 B 样条扭曲变形

Require: 导引图的 68 个关键点 P_i, 源图的 68 个关键点 $P_i^{'}$
Ensure: 变形后的图像

function 三次 B 样条扭曲变形 (x', y', x, y)
    做 (x', y') 到 (x, y) 的仿射变换实现特征点对齐
    以源图的宽高进行网格的建立
    计算控制点的位移，即 $(x' - x, y' - y)$
    以权重为 1，计算每个网格点的位移
    初始化位移权重、LearningRate 并进行梯度下降更新参数
    通过任一点的周围 16 个点的位移来计算该点的位移
    ImgDst.SetPixel 并进行插值。
return ImgDst

4 68 个关键点检测

使用 $C^{\#}$ 的 Dlib 库运用机器学习进行关键点检测[1]。在加载已经训练好的模型后，检测即可得到 68 个关键点。通过与附件中给定的 9 张图的关键点坐标相对比，相差较小。

5 插值原理

5.1 最近邻插值

将变换后的图像中的元像素点最近邻像素的灰度值赋给原像素点的方法，是最简单的灰度值插值，即将非整点的坐标四舍五入，不够精确。

5.2 双线性插值

图像中任意一点的像素值由它周围四个点的像素值确定，具体关系为

$$f(i+u, j+v) = [1-u \ u] \begin{bmatrix} f(i,j) & f(i,j+1) \\ f(i+1,j) & f(i+1,j+1) \end{bmatrix} \begin{bmatrix} 1-v \\ v \end{bmatrix}$$

双线性插值相对简便，速度相对较快。但是双线性灰度插值的平滑作用可能使得图像的细节产生退化，这种现象在进行图像放大时尤其明显。

5.3 双三次插值

图像中任意一点的像素由它周围 16 个点的像素确定，具体关系为

$$f(i+u, j+v) = ABC^T$$

其中，矩阵 A、B、C 分别为

$$A = [S(u+1) \ S(u) \ S(u-1) \ S(u-2)]$$
$$B = f(i-1:i+2, j-1:j+2)$$
$$C = [S(v+1) \ S(v) \ S(v-1) \ S(v-2)]$$

S(•) 为

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

[1]参考了 Github 上有关 dlib 的 $C^{\#}$ 相关开源代码

双三次插值通常能够产生较好的效果，最精确的插补图形，能够创造出比双线性插值更平滑的图像边缘。但是它的速度几乎是最慢的。

6 实现效果

6.1 运行界面

用户可以根据需要对变形方式进行选择 (TPS/BSpline)，对三种插值方式进行选择。同时对于关键点信息的载入，用户可以选择读入外部数据或者选择自动识别。同时，为了防止用户操作上的失误，设置只有在用户载入图片之后才能上载关键点信息，二者均完成后才能开始进行变形。

图 1: 运行界面

6.2 不同变形模式下的运行效果

6.2.1 TPS 变形

上图即为 TPS 变形的效果。由于 TPS 的原理，故在变形之后会出现没有信息的部分，对于该部分的处理为忽略。之前考虑用最边缘的像素值进行 padding，但是由于本身就缺乏该部分信息，进行 padding 之后图像会变得很奇怪。

6.2.2 B 样条变形

在进行 20 次梯度下降之后，得到变形后的图片。由于 B 样条只对局部进行处理，所以背景上没有缺失信息的部分。如这张图，奥巴马的眼睛变大。

图 2: B 样条变形界面

6.3 不同插值方式下的变形效果

以奥巴马的脸特征换成 Anderson 为例，使用 TPS 变形方式。大图见 figure 文件夹。由于看小图无法感受到插值效果的不同，故将图片放大 5 倍对比奥巴马的右眼。

6.3.1 最近邻插值

图 3: 最近邻插值效果图

6.3.2 双线性插值

图 4: 双线性插值效果图

6.3.3 双三次插值

图 5: 双三次插值效果图

6.3.4 三种插值方式效果对比

由对比图可以看到，最近邻插值的效果最差，双线性和双三次插值的效果要好于最近邻插值。除此之外，双三次插值的图像边缘要比双线性平滑，即双线性插值可能会对图像产生稍微模糊的效果。

图 6: 三种插值方式效果对比，从左到右分别为最近邻插值、双线性插值、双三次插值

6.4 自行检测关键点并进行变形

在实现自行检测 68 个关键点之后，可以对想要被变形的图片进行变形。下图尝试“王校长”的变形，68 个关键点等在检测结束后自行显示在图片上。在识别关键点之后，可以得到变形后的图片。可以看到，变形后的“王校长”更符合图片的配字。

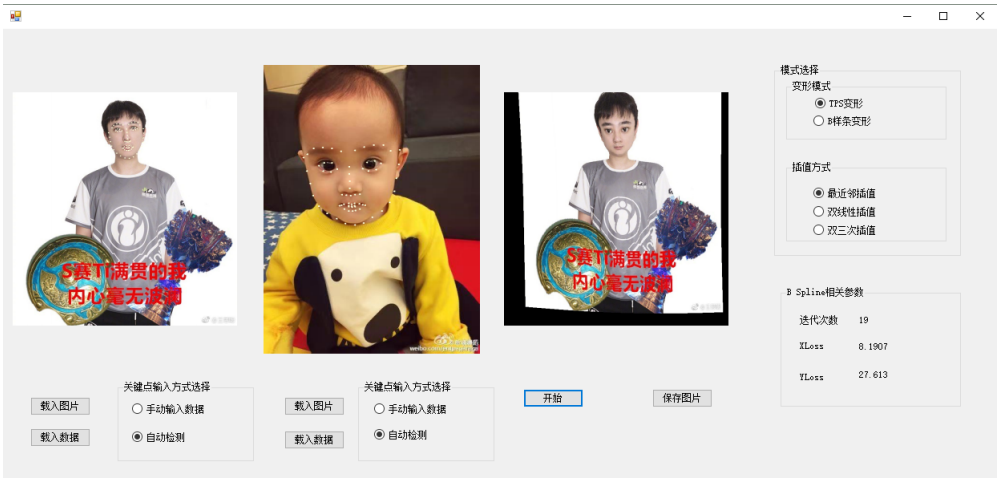

图 7: 自行检测关键点变形

识别算法可以识别这种通过人脸制作的表情包。

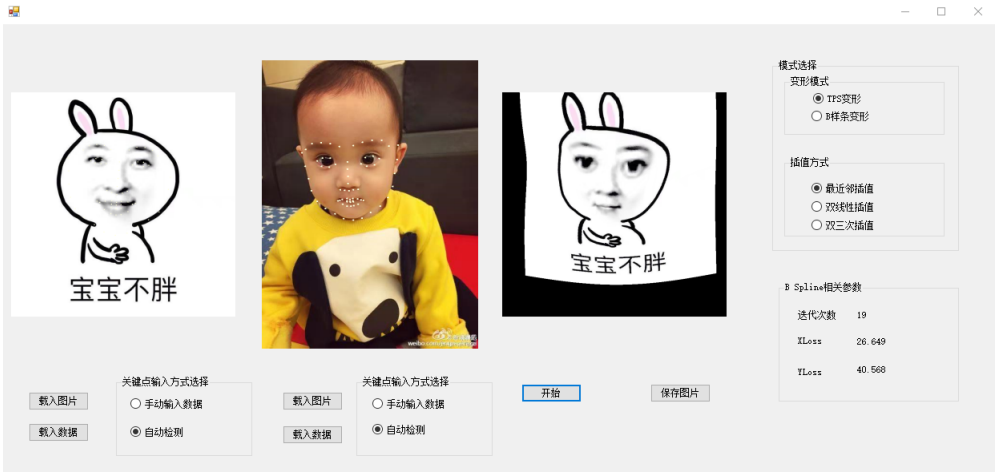

图 8: 自行检测关键点变形

7 误差分析

7.1 观测误差与舍入误差

对于观测误差，由于图片的 RGB 值是 uint8 类型，所以观测误差最大为 0.5。

工程中多出使用 $C^{\#}$ 的 double 类型，精度为 15 位有效数字，故中间过程的舍入误差相对较小，可以忽略。而在对 Bitmap 的 RGB 赋值时，将 double 类型转换为 0 255 的 uint8，舍入误差最大为 0.5。而对于这三种插值，其实质为周围 4 个或 16 个点的线性组合，且线性组合的系数为 1。

综上，观测误差的最大值为 0.5，舍入误差的最大值为 0.5。

7.2 方法误差 (插值)

7.2.1 最近邻插值

由坐标的舍入误差可以得到，$|\Delta x| \le 0.5, |\Delta y| \le 0.5$，所以最近邻的方法误差为

$$\begin{aligned} |\Delta A| &\le \max\left(\left|\frac{\partial f}{\partial x}\right|\right)|\Delta x| + \max\left(\left|\frac{\partial f}{\partial y}\right|\right)|\Delta y| \\ &\le \frac{1}{2}\left[\max\left(\left|\frac{\partial f}{\partial x}\right|\right) + \max\left(\left|\frac{\partial f}{\partial y}\right|\right)\right] \end{aligned}$$

由于数字图像为离散信号，故将微分换位差分。假定，$f(x,y)$ 在 x 方向和 y 方向上的一阶导数存在且有界为 M_{x1}, M_{y1}。则显然，在物体边缘等变化较大的区域，M_{x1}, M_{y1} 比较大，其他区域非常小。所以，x 方向和 y 方向的方法误差最大为 $0.5M_{x1}, 0.5M_{y1}$，总方法误差为 $0.5(M_{x1} + M_{y1})$。

7.2.2 双线性插值

假定，$f(x,y)$ 在 x 方向和 y 方向上的二阶导数存在且有界 M_{x2}, M_{y2}。先考虑 x 分量的插值，即考虑 $y = j$ 与 $y = j+1$ 方向的插值。每个直线方向上的插值为一维插值，故可以得到每条直线的插值误差为

$$|R_1(x,y)| \le \frac{M_{x2}}{2}\max x(1-x) \le \frac{M_{x2}}{8}$$

再将得到的插值后的两点进行 y 分量的插值，即考虑 $x = x_0$ 方向的插值，且同为一维插值，故可以得到插值误差为

$$|R_1(x,y)| \le \frac{M_{y2}}{2}\max y(1-y) \le \frac{M_{y2}}{8}$$

综上，总的方法误差为 $\frac{M_{x2}+M_{y2}}{8}$。

7.2.3 双三次插值

假定，$f(x,y)$ 在 x 方向和 y 方向上的四阶导数存在且有界 M_{x4}, M_{y4}。若直接按照分析双线性插值对双三次插值误差进行分析，则很容易得到总的方法误差为

$$R(x,y) \leq \frac{3}{128}(M_{x2}+M_{y2})$$

考虑双三次插值的另一种形式[2]。具体如下

$$p(x,y) = [1, x, x^2, x^3] \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} 1 \\ y \\ y^2 \\ y^3 \end{bmatrix}$$

其中，参数矩阵可以表示为

$$\begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix} = P^T F P$$

其中，矩阵 P 和矩阵 F 分别为

$$P = \begin{bmatrix} 1 & 0 & -3 & 2 \\ 0 & 0 & 3 & -2 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & -1 & 1 \end{bmatrix} \qquad F = \begin{bmatrix} f(0,0) & f(0,1) & f_y(0,0) & f_y(0,1) \\ f(1,0) & f(1,1) & f_y(1,0) & f_y(1,1) \\ f_x(0,0) & f_x(0,1) & f_{xy}(0,0) & f_{xy}(0,1) \\ f_x(1,0) & f_x(1,1) & f_{xy}(1,0) & f_{xy}(1,1) \end{bmatrix}$$

记 P_x, P_y 矩阵如下

$$P_x = P \begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \end{bmatrix} \qquad P_y = P \begin{bmatrix} 1 \\ y \\ y^2 \\ y^3 \end{bmatrix}$$

则有

$$p(x,y) = P_x^T F P_y$$

根据这个形式可以得知，双三次插值即可以等价位，先做 y 方向的三次埃尔米特插值，再做 x 方向的三次埃尔米特插值，即与双线性插值分解方法类似。则误差为 $\frac{1}{384}(M_{x4}+M_{y4})$。

而在求取导数的过程中同样存在着误差。由于离散情况下，需要使用差分来代替导数，考虑使用中心差分来近似一阶导数。具体形式如下，这里 h 取 1。

$$f^{'}(x_i) = y^{'}{}_i \approx \frac{y_{i+1}-y_{i-1}}{2h}$$

在这里，有

$$f_x(x,y) \approx \frac{1}{2}[f(x+1,y) - f(x-1,y)]$$

$$f_y(x,y) \approx \frac{1}{2}[f(x,y+1) - f(x,y-1)]$$

[2]该形式借鉴于维基百科，双三次插值。链接为 https://en.wikipedia.org/wiki/Bicubic_interpolation

不妨先考虑 x 方向，则易知存在 $x^* \in [x-1, x+1]$ 使得 $f_x(x) = f'(x^*)$。假定 $f(x,y)$ 的二阶导数存在且有界 M_{x2}, M_{y2}。所以有

$$|R(x,y)| = |f^{'}(x^*) - f^{'}(x)| \leq M_{x2}|x^* - x| \leq M_{x2}$$

y 方向同理。所以由于使用差分来代替导数造成的误差为 $M_{x2} + M_{y2}$。双三次插值总的方法误差为二者相同时考虑。

7.3 方法误差 (变形)

7.3.1 最小二乘拟合

最小二乘参数拟合使用了高斯消元法进行求解，具体分析见下。

7.3.2 TPS 变形

高斯消元法 线性方程组 $L\left[\omega_1, ..., \omega_n, a_1, a_x, a_y\right]^T = Y$ 求解时应用了高斯消元法。由定理[3]，若 A 是非奇异阵，$Ax = b \neq 0$，且 $A(x + \delta x) = b + \delta b$，则有

$$\frac{\|\delta x\|}{\|x\|} \leq \|A^{-1}\| \ \ \|A\| \ \ \frac{\|\delta b\|}{\|b\|}$$

高斯消元误差的上界即为 $\|A^{-1}\| \ \ \|A\|$, 经过 Matlab 辅助计算，所给定的 9 份数据点 (最小二乘法拟合后) 的所构成的矩阵 L 不是病态矩阵。但是由于水平有限，无法进行更详细的分析。
除此之外，舍入误差与观测误差没有被放大，几乎无影响。

7.3.3 B 样条变形

梯度下降法 迭代次数限制在 20 次时，根据当前实现的算法，代价函数 (均方误差) 的区间为 (0,40)。而当参数最佳时，均方误差为 $E(E \geq 0)$，故梯度下降法的均方误差为 $(0, 40)$。

8 其他相关说明

8.1 TPS 与 B 样条的对比

通过视觉效果来看，TPS 是针对全局进行变形，而 B 样条是对局部进行变形，所以论全局观赏效果的话，B 样条占优势。但是显而易见 TPS 的变形效果要比 B 样条更明显。除此之外，B 样条的计算时间要比 TPS 少一些。

8.2 问题与解决

8.2.1 关于线性方程组求解的误差分析

由于自身能力有限，关于高斯消元法部分的误差分析在稍微自学课本后面的内容之后，仍是无法进行更具体的分析。我感觉可能无法去分析得更具体，即若分析误差上界，可能实际值要远小于该误差上界，故或许需要一个条件更强的分析高斯消元误差的方法。

8.2.2 关于 LOSS 函数的选取

LOSS 函数的选取本着网格点与移动点的位移关系设计，但事实证明若 LOSS 函数趋于 0 所得到的变形并不是最优的，猜测设计的 LOSS 函数的条件不够强。为了解决这种情况，将迭代次数固定。

[3]参照数值分析教材 168 页定理 21

8.2.3 关于图片中有多个人

68 个关键点检测算法是先对人脸进行识别，再载入模型进行检测关键点。如果图片中有多个人，则默认人脸为所检测的最后一个人脸。

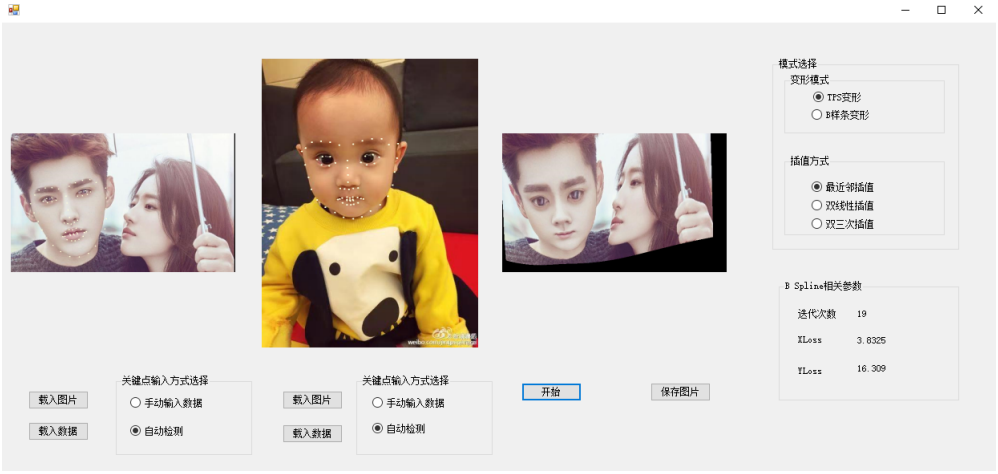

图 9: 多人图片实现效果

8.2.4 关于侧脸和歪脸的处理

变换最好的效果就是正脸源图和正脸导引图。如果有歪脸，即头正，则通过仿射变换最小二乘，已经进行旋转，故效果较好。但是对于侧脸，比如梁静茹和特朗普的另一张照片，变换之后的图像为了模仿导引图的特征，脸会歪，但是这是正确的输出结果。同时由于没有露出的脸部的信息未知，所以无法对未知的信息进行填补。

8.3 程序运行所需环境

程序使用 VS2017 进行开发，使用的语言为 $C^{\#}$，电脑分辨率为 1366×768。

8.4 总结与反思

本次大作业，让我慢慢地了解并应用了 $C^{\#}$ 语言，并让我体会到了它的优点所在。除此之外，对插值的理解更加深刻。而对于图像的变形上，我感受到了 TPS 变形与 B 样条变形之间的差异。TPS 相对来讲实现简单，但是对全局进行操作导致背景出现黑色部分；而 B 样条可以对局部进行调整，但是调整效果并没有 TPS 那么明显，各有利弊。以及由于还有其他几个大作业，关于 B 样条还有一些升级的思路，但是由于时间关系没有付诸实践。除此之外，由于时间关系，没有对界面进行美化，保持着原始的朴素。

此外，在逐渐完成大作业的过程中，我发现我之前将换脸和脸部变形的概念混淆。本次大作业完成的只是根据导引图关键点特征对源图进行变形，生成图片的像素值采自源图，所以无法实现换脸的效果。