数字图像处理 Project 1

计 76 陈之杨 2017011377

2020.4

1 Image Fusion

1.1 原理

假设我们要将 A 图的 Ω 区域融合到 B 图中。设 A 图的颜色为 g，融合后的颜色为 f，B 图的颜色为 f^*。由于我们可以将 RGB 三个通道分别处理，这里把 f, g, f^* 看作单值函数。

我们希望 f 的边界能够和 f^* 一致，但是 f 的内部梯度接近 g 从而保留 A 图的语义信息。于是可以求解如下的优化问题：

$$\min_f \iint_\Omega \|\nabla f - \nabla g\|^2,$$
$$\text{subj. to } f\big|_{\partial\Omega} = f^*\big|_{\partial\Omega}.$$

这个优化问题有封闭形式的解

$$\Delta f = \Delta g, f\big|_{\partial\Omega} = f^*\big|_{\partial\Omega}.$$

也就是边界值固定的线性方程组。注意到每个方程只与对应像素和它的相邻 4 个像素有关，因此这个方程组是稀疏的，可以使用稀疏方程组求解（对应 `scipy` 库里的 `sparse.linalg.spsolve`），从而提高效率。

但是，直接求解泊松方程，可能会导致目标图的纹理信息被覆盖，所以可以对泊松方程作如下的修改：

$$\min_f \iint_\Omega \|\nabla f - v\|^2,$$

其中 v 取 ∇f^* 和 ∇g 中长度较大的一个。这样融合时优先保持梯度变化剧烈的部分，从而保护原图的纹理信息。

1.2 实验结果

见图 1。

保持纹理的结果如图 2所示。

图 1: 泊松图像融合。

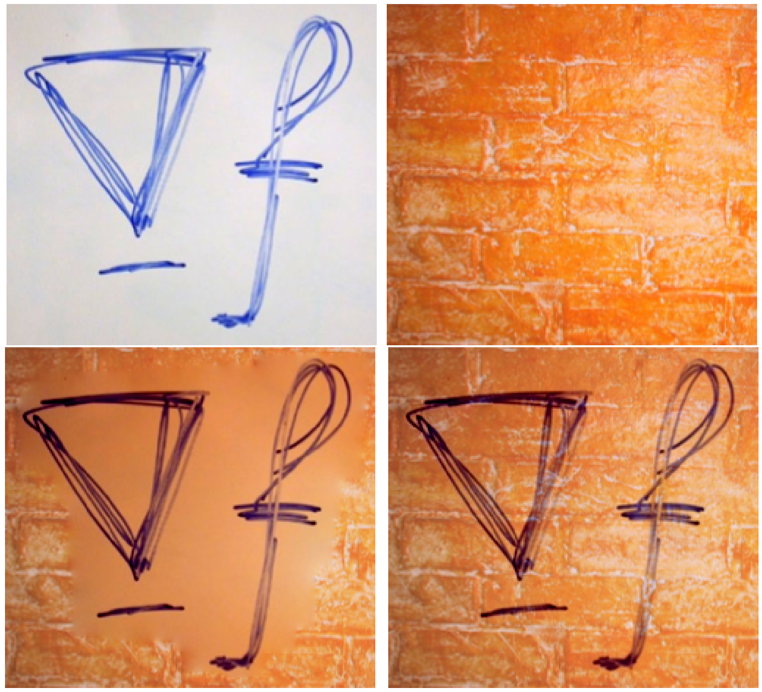

图 2: 保持纹理信息前后的结果。

2 Image Morphing

2.1 原理

假设我们希望得到两张图之间的平滑过渡，并且两张图的相同的结构特征可以保持。于是我们可以首先选定两张图的特征点对，并以特征点为顶点，求图片的三角剖分。之后，再对剖分图的每个三角形分别进行连续的线性插值。这样就可以保持图片变换时的结构信息。

实现时，求解 Delaunay 三角剖分使用了 `python` 的 `scipy.spatial.Delaunay` 库。

2.2 实验结果

见图 3。特征点对为手动标注。

3 View Morphing

3.1 原理

普通的 Image Morphing 只考虑了图片的结构对应关系，但没有考虑不同图片的视角差异。我们希望可以由两张不同视角的图片得出视角变换的平滑过渡。

View Morphing 分为三步：预变换，根据特征点对的关系，通过透视变换将两张图片变形为

图 3: Image Morphing.

平行视角；变形，使用普通的 Image Morphing 生成过渡图片；后变换，将得到的图片视角变换回非平行视角。下面说明如何进行预变换和后变换。

假设两张图片 I_0 和 I_1 有一对对应的特征点 p_0 和 p_1，根据对极几何，存在一个基本矩阵 F，使得

$$p_1^{\mathrm{T}} F p_0 = 0,$$

这里使用的是齐次坐标，因此 $F \in \mathbb{R}^{3\times 3}$，且 $\operatorname{rank}(F) = 2$。基本矩阵描述了两张图片之间的变换关系。求基本矩阵可以看作一个最小二乘问题，为了减小误差，通常取多对特征点，使用 RANSAC 算法求解。由于我自己实现的求基本矩阵算法误差太大，无法得到满意的解，这里使用 `opencv` 的 `findFundamentalMat` 函数求解。

为了对齐两张图片，使得视角平行，需要进行预变换。可以证明，当两张图片平行时，基本矩阵有如下的形式：

$$\hat{F} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}.$$

因此，预变换的实质是找到变换 H_0 和 H_1，使得下式成立：

$$(H_1^{-1})^{\mathrm{T}} F H_0^{-1} = \hat{F}.$$

图 4: Image Morphing.

此时有

$$(H_1 p_1)^{\mathrm{T}} \hat{F} (H_0 p_0) = 0.$$

对于 I_0 中的一条极线 $d_0 = [d_0^x,\ d_0^y,\ 0]^{\mathrm{T}}$，线上的点 $Fd_0 = [x,\ y,\ z]^{\mathrm{T}}$ 在 I_1 上对应的极线为 $d_1 = [-y,\ x,\ 0]^{\mathrm{T}}$。根据极点信息可以求出图片的旋转角。在把图像变为平行视角后，还要旋转图片使特征点对齐。这里只需要计算变换后的极点 $\hat{e} = [\hat{e}^x,\ \hat{e}^y,\ \hat{e}^z]$，于是旋转角度为

$$-\arctan\left(\frac{\hat{e}^y}{\hat{e}^x}\right).$$

论文里没有给出后变换的计算方法，不过我们可以直接根据预变换后的四个角点，将它们重新拉伸到图像四角即可。

3.2 实验结果

见图 4。由于给定的图片不是很清晰，计算的结果质量不是很高。