

数字图像处理综合作业四——图像替换

陈昭熹 2017011552

2020 年 1 月 14 日

目录

1 运行说明	3
1.1 ./task1	3
1.1.1 油画风格变换	3
1.1.2 卡通风格变换	3
1.1.3 老照片风格变换	3
1.1.4 薄板样条变形 (TPS)	3
1.1.5 模板处理与图像替换入口	3
1.1.6 所使用的模板图像	4
1.2 ./task2	5
2 算法流程与实现	5
2.1 任务一——单帧图像替换	5
2.1.1 基于 Bayesian Matting 的图像分割	6
2.1.2 风格变换	8
2.1.3 基于 TPS 的配准与变形	12
2.2 任务二——视频图像替换	13
2.2.1 SURF 特征提取与筛选	13
2.2.2 基于 Ransac 求解仿射变换矩阵	14
2.2.3 基于优化求解仿射变换矩阵	14
2.2.4 基于局部梯度方向的角点更新	14
2.2.5 基于 Hough 直线检测的角点更新	16

目录	2
----	---

3 中间过程及结果	18
------------------	-----------

3.1 任务一	18
3.2 任务二	24

1 运行说明

由于本次作业代码量较大，同时分成两个小任务，因此将相关文件分到两个目录中，任务一对应 task1，任务二对应 task2。

1.1 ./task1

1.1.1 油画风格变换

painting_trans.m 油画风格变换函数

SLIC_Proc.m 超像素分割，用于油画风格变换

adaptive_ks.m 自适应计算超像素个数 ks

1.1.2 卡通风格变换

carton_trans.m 卡通风格变换函数

1.1.3 老照片风格变换

oldpic_trans.m 老照片风格变换函数

1.1.4 薄板样条变形 (TPS)

morph_tps_wrapper.m TPS 变形函数入口

morph_tps.m TPS 主逻辑

morph.m 图像变形

est_tps.m TPS 参数矩阵求解

pixel_limit.m 图像边界约束，防止像素点越界

1.1.5 模板处理与图像替换入口

对应 6 张模板图像，准备了六个处理脚本，可以直接运行脚本查看效果。脚本命名为 for_target*i*.m，其中 $i = \{1, 2, 3, 4, 5, 6\}$

1.1.6 所使用的模板图像

除去要求的三张模板外，自定义的三张模板如下所示：



图 1：自选模板 1



图 2：自选模板 2



图 3: 自选模板 3

1.2 ./task2

在 outputs 文件夹下放置了追踪并替换掉封面内容后的图像序列，同时将这些图像转换成了 result.avi 视频文件供查阅。pics 文件夹下为原始视频转出的图像序列。

video_image_scripts.m 视频转图像序列 or 图像序列转视频

HoughRefine.m 基于 Hough 直线检测的角点优化

RansacFitHomography.m Ransac 方法求解单应性矩阵

ImageTracking.m 追踪四个角点，并实时替换封面内容

main_entrance.m 任务二入口脚本

2 算法流程与实现

2.1 任务一——单帧图像替换

任务一整体工作流程如下所示：

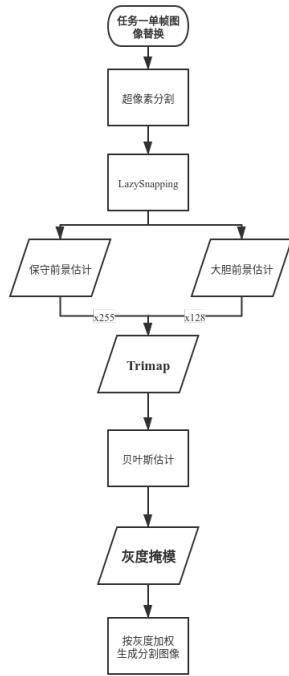


图 4: 任务一流程图

2.1.1 基于 Bayesian Matting 的图像分割

首先利用综合作业三中基于超像素的懒人切图方法得到粗分割的两类 mask, 其中一种是置信度较高的, 估计较为保守的, 但一般前景范围较小; 另一种是置信度较低的, 估计较为大胆的, 但一般前景范围较大。两种 mask 在前景区域的差集即在前背景过渡的不确定区域。利用两种 mask 生成 trimap, 即包含三种灰度值 (0,128,255) 的 mask 供图像分割使用。

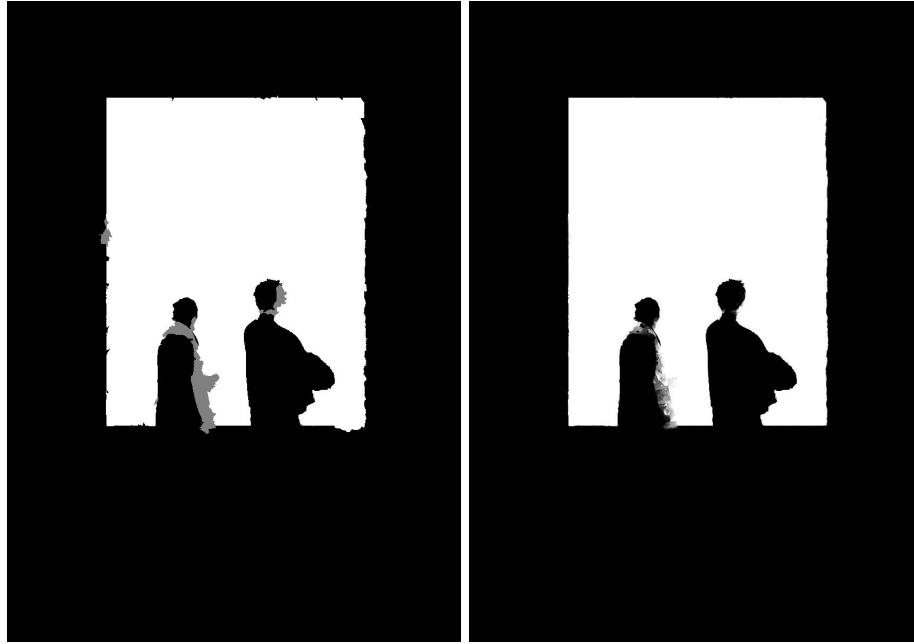


图 5: 模板 1 的 trimap 与灰度 mask

在图像分割阶段，使用基于贝叶斯估计的方法 [1]，即利用上述 trimap，利用彩色图像内的信息进行前景概率估计，最终生成灰度掩模 α ，其中灰度值越高则代表前景置信度越大，反之则更可能为背景。利用下式即可获得过渡较为自然的抠图效果：

$$I = \alpha F + (1 - \alpha)B \quad (1)$$

其中 I 为目标图像， F 为前景， B 为背景，在本任务应用场景中即 F 为源图像， B 为模板图像。在本文代码实现过程中，对于权值进行了缩放以便适应于不同的风格与模板，公式如下：

$$I = \alpha^k F + (1 - \alpha^k)B \quad (2)$$

其中 k 为缩放因子，在实验过程中一般取 3-10 的整数，取决于在过渡区域想要实现的自然效果而定。

2.1.2 风格变换

本次实验最终实现了五种风格变换，分别是油画风格、老照片风格、卡通风格、彩色素描风格以及风格相似化。

油画风格

受到课上老师在超像素分割部分讲解的启发，通过对源图进行超像素分割，并将每一个超像素内的原像素设置为类内均值来模拟油画中的块状效果。



图 6: 模板 3 的油画效果

老照片风格

通过对原照片在 RGB 色彩空间内将使用三通道的加权值对每一个通道进行处理，最终获得照片变暗同时色调发黄的效果。若变换前三通道色彩值表达为向量 $[R \ G \ B]$ ，变换后表达为 $[R' \ G' \ B']$ ，则变换关系如下：

$$[R' \ G' \ B']^T = \begin{bmatrix} 0.393 & 0.769 & 0.189 \\ 0.349 & 0.686 & 0.168 \\ 0.272 & 0.534 & 0.131 \end{bmatrix} [R \ G \ B]^T \quad (3)$$

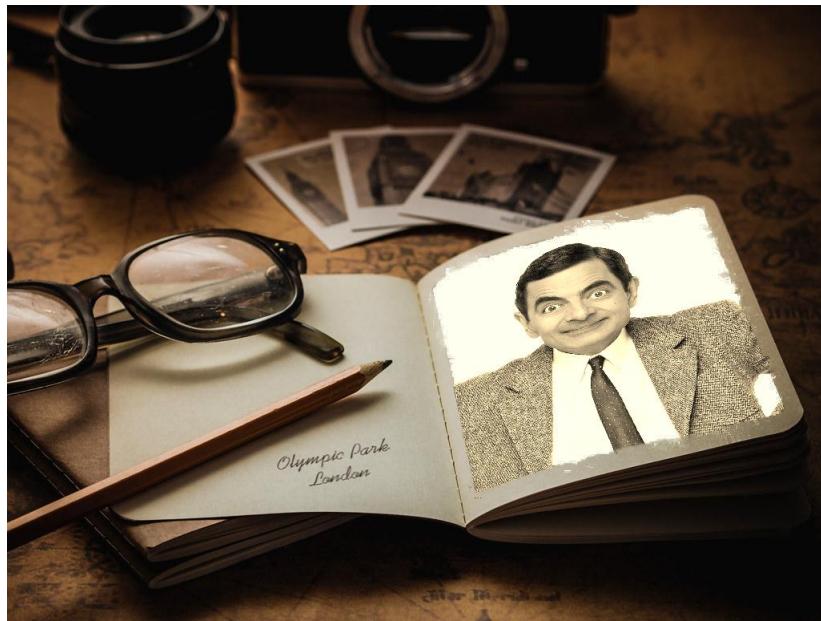


图 7: 模板 4 的老照片效果

卡通风格

利用双边滤波模糊图像平坦部分，并保留边缘，来实现卡通画中的不真实的块色彩。同时使用 canny 算子求解边缘，将其加到原图中，来模拟卡通画中的残留笔触、褶皱等效果。



图 8: 模板 1 的卡通风格

彩色素描风格

本部分变换参考了港中文的工作 [2]，通过对原图进行滤波处理，同时将几种已知的铅笔笔触视作纹理特征融入到原图片中。



图 9: 模板 1 的彩色素描风格

风格相似化

风格相似化作为一个基础功能并没实现为函数，由于任务二中也同时需要这样的功能，因此直接以代码段的形式插入到了需要的地方。其思想并不复杂，即在 HSV 色彩空间每一个通道上对源图和目标图进行直方图匹配(imhistmatch)，其中目标图作为参考直方图，将变换后的直方图通过权重加到源图上，从而实现风格相似化处理。



图 10: 模板 6 中的风格相似化

2.1.3 基于 TPS 的配准与变形

在本任务中之所以选择算法复杂度更高的 TPS 而非仿射或射影变换，是为了利用 TPS 的平滑变形效果来达到少量控制点就能得到平滑边缘的目标。例如在模板 2 中，目标区域的上边缘是光滑的曲线而非直线，这在利用线性变形时会产生较大的误差，而 TPS 辅以灰度掩模则可以取得较好的效果。

TPS 即薄板样条变形，其主要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，能量函数定义与求解过程无关，在此不进行赘述。下面给出 TPS 求解过程：给定 n 个控制点

$$\{P_1 = (x_1, y_1), \dots, P_n = (x_n, y_n)\}$$

假设目标点为

$$\{\tilde{P}_1 = (x'_1, y'_1), \dots, \tilde{P}_n = (x'_n, y'_n)\}$$

定义如下辅助变量：

$$K = \begin{bmatrix} 0 & U(r_{12}) & \dots & U(r_{1n}) \\ U(r_{21}) & 0 & \dots & U(r_{2n}) \\ \dots & \dots & \dots & \dots \\ U(r_{n1}) & U(r_{n2}) & \dots & 0 \end{bmatrix} \quad (4)$$

$$P = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ \dots & \dots & \dots \\ 1 & x_n & y_n \end{bmatrix} \quad (5)$$

$$L = \begin{bmatrix} K & P \\ P^T & \mathbf{0} \end{bmatrix} \quad (6)$$

$$V = \begin{bmatrix} x'_1 & x'_2 & \dots & x'_n \\ y'_1 & y'_2 & \dots & y'_n \end{bmatrix} \quad (7)$$

$$Y = \begin{bmatrix} V_1 & 0 & 0 & 0 \\ V_2 & 0 & 0 & 0 \end{bmatrix}^T \quad (8)$$

$$L [w_1, \dots, w_n, a_1, a_x, a_y]^T = Y \quad (9)$$

则所拟合出的光滑曲面（即一个关于 xy 的二元函数，函数值对应着像素点的像素值）：

$$f(x, y) = [f_x(x, y), f_y(x, y)]^T = a_1 + a_x x + a_y y + \sum w_i U(|P_i - (x, y)|) \quad (10)$$

值得注意的是，其中定义了径向基函数

$$U(r) = \begin{cases} r^2 \log(r^2), & r \neq 0 \\ 0, & r = 0 \end{cases}$$

这个函数直观的给出了控制点周围的变形插值函数，不同的距离体现了不同的变形效果。

在本任务中，使用四到五个角点作为控制点进行变形。

2.2 任务二——视频图像替换

视频中的图像替换采用以下流程：

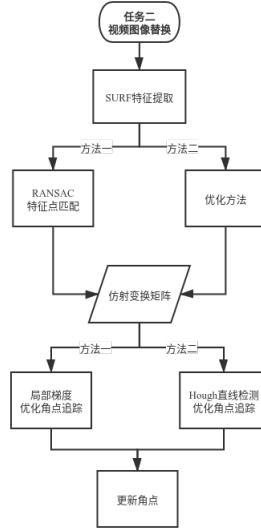


图 11: 任务二流程图

2.2.1 SURF 特征提取与筛选

特征点选择使用计算快速且较为鲁棒的 SURF 特征，并限制特征点个数以保证计算速度。利用 `detectSURFFeatures` 函数进行提取。在特征点提取结束后，利用其度量距离作为特征质量，筛选出质量较高的点作为匹配点，用于计算相邻帧之间的变换矩阵。

2.2.2 基于 Ransac 求解仿射变换矩阵

本节介绍特征点追踪的其中一种方法，基于 Ransac 的方法。利用得到的匹配特征点可以计算相邻两帧之间的变换矩阵。在观察过目标视频后发现，待追踪的目标为书籍封面，且相机畸变较少，封面的对边在相邻帧保持了平行关系，为了降低系统自由度，选择了保平行关系的仿射变换。事实上，在实验过程中，使用射影变换往往会导致更差的结果。这是由于 8 自由度的约束在本任务低自由度的运动下会误差更大。这也从侧面证明了，要具体问题具体分析，不一定越复杂的方法越好，适合的方法才是最好。为了提高相邻帧变换关系的准确度，使用 RANSAC 方法来筛选最优的特征点对来计算仿射变换矩阵，最优性由残差决定，残差定义为匹配特征点对坐标差距的二

范数:

$$error = ||ref - dst||_2 \quad (11)$$

通过随机选点的方法以及足够的迭代次数保证从匹配点中选择到让变换误差最小的一组特征点对，从而获得相邻帧之间的仿射变换矩阵，用于追踪角点。

2.2.3 基于优化求解仿射变换矩阵

本节介绍特征点追踪的另一种方法，基于优化的方法。利用库函数 estimateGeometricTransform，将所有提取到的匹配点均作为已知量，将相邻帧之间的变换矩阵求解从线性方程组转变为超定方程的优化问题。经过实验，该方法由于约束更多更严格，因此获得的效果更好，在平移过程中追踪的角点基本不会发生漂移，无需额外修正，但计算代价从期望的角度上比上一节的方法要高，因而运行速度会变慢。

2.2.4 基于局部梯度方向的角点更新

本节介绍角点更新的其中一种策略。由于仿射变换矩阵保证特征点对的误差小，但不能保证全局下每一个像素点都是准确的，因此对于待追踪的四个角点在追踪不正确的情况下需要使用额外的方法进行位置修正，以保证追踪位置一直固着在书籍封面的四角。

注意到书籍封面主体颜色为蓝色，桌面背景色为黄色，因此在书封面的边缘会产生较大的梯度，利用这一点就可以很好的确定角点位置。使用 canny 算子求得原图的边缘图：



图 12: canny 算法的边缘二值图

再求出边缘图的梯度幅度和方向:



图 13: 边缘图梯度幅度 & 梯度方向

由于梯度方向一定程度上表征了图像边缘的走向，因此在角点追踪误差不大的情况下，在未修正的角点附近沿着梯度方向前进一定能够找到角点。具体做法是，沿着梯度方向前进直至梯度方向有 90 度以上的改变则判定为真实角点。实现过程中，采用在未修正的角点邻域内（取 $[15 \times 15]$ ）找极差的方法。最终角点位置取为极差点的中心：

$$refined = \frac{P_{min} + P_{max}}{2} \quad (12)$$

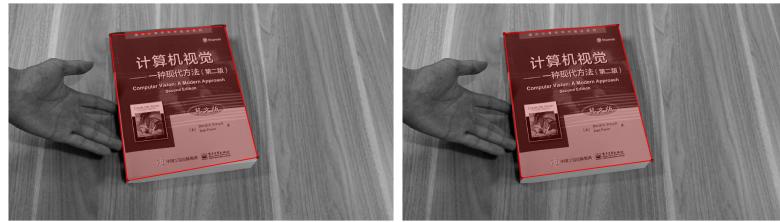


图 14: 局部梯度方向修正角点前后

实验过程中，这样的更新策略计算量较少，但建立在未修正的角点也是比较准确的前提之上。因此这一策略适用于配合着该方法2.2.3使用。

2.2.5 基于 Hough 直线检测的角点更新

本节介绍角点更新的另一策略。利用先验信息——书的封面边缘是直线，因而角点必在直线的交点处。通过 Hough 变换检测直线的方法（由于之前的作业已经做过在此不赘述），检测图像内的直线，通过直线方程匹配的方法找到所需要的四根直线，并计算出交点作为更新的角点。需要注意的是，在检测直线时应当使用 LAB 色彩空间下的第三通道进行边缘提取，这样做的目的是利用该通道下前景与背景的较大差异，尽量能够提取出高质量的书籍封面边缘，同时减少封面内部文字和纹理带来的干扰，增加高质量直线、高概率解出现的可能。同时应当在检测时设置直线最小长度阈值（实验中使用的是 15），这样能够规避噪声同时减少匹配次数，提高计算速度。在寻找最优匹配直线时，由残差进行约束。将未修正的四个角点首尾相连计算出四条直线 l_1, l_2, l_3, l_4 ，待匹配的最优直线为 l ，则对于直线 l_i ，残差定义如下：

$$\text{error} = (k_i - k)^2 + 0.001 \times (b_i - b)^2 \quad (13)$$

在所有直线中找残差最小的即匹配的直线，可以得到类似下面的结果。

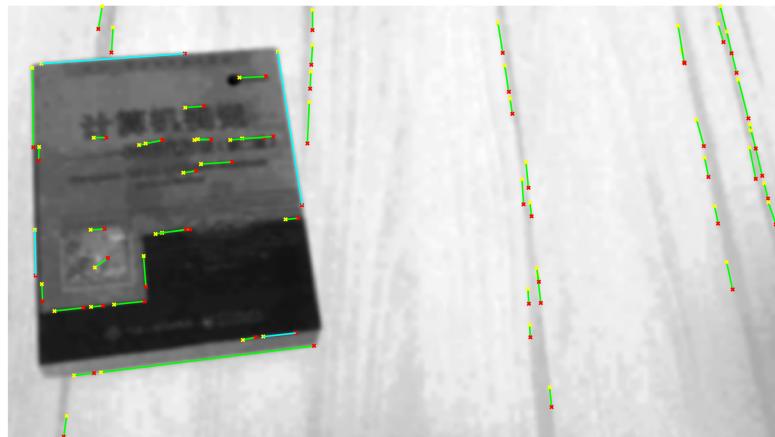


图 15: 视频前期直线匹配结果

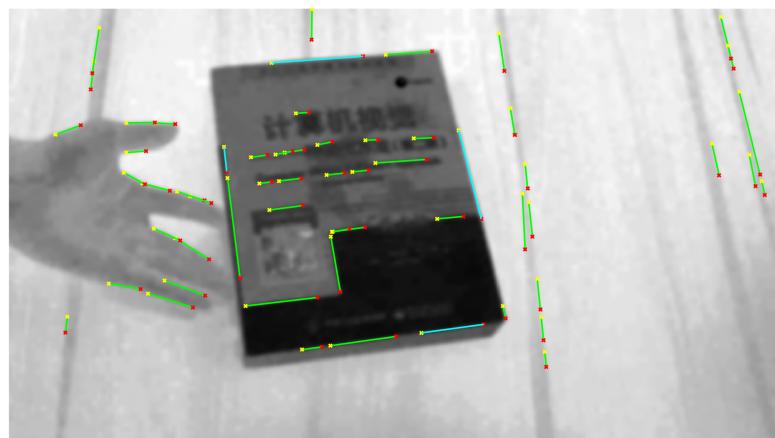


图 16: 视频中期直线匹配结果

找到对应封面边缘的四条最优匹配直线后，直接通过解线性方程的方法解出交点即更新后的角点位置。

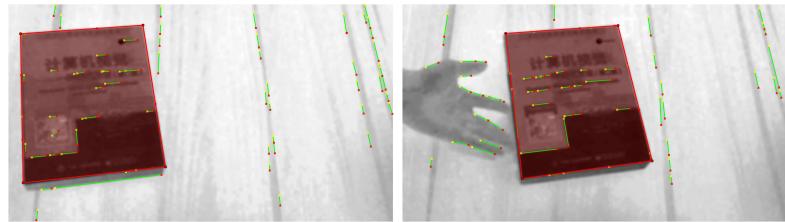


图 17: 通过交点更新角点

3 中间过程及结果

3.1 任务一

下面给出任务一中的最终效果图:



图 18: 模板 1——卡通风格



图 19: 模板 1——彩色素描风格



图 20: 模板 2



图 21: 模板 3——油画风格



图 22: 模板 4——老照片风格



图 23: 模板 5——报纸风格



图 24: 模板 6——粉刷风格

下面是展示所有模板图像的图像分割效果:

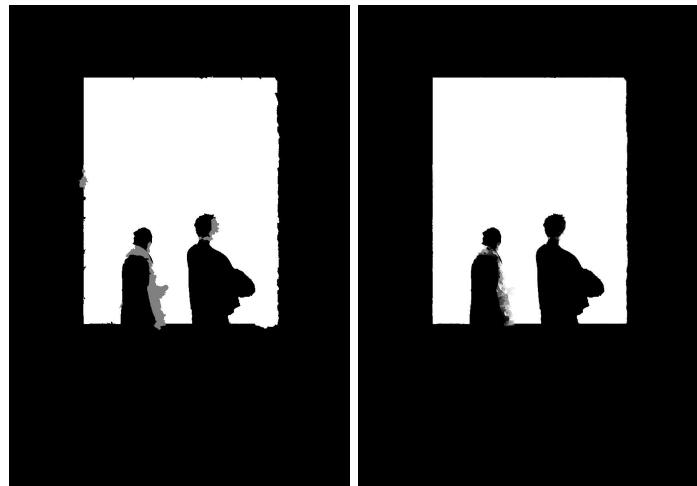


图 25: 模板 1 的 trimap 和灰度掩模

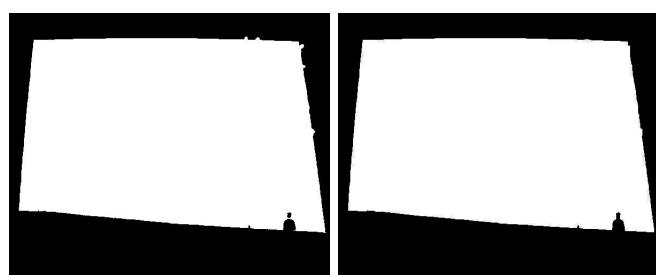


图 26: 模板 2 的 trimap 和灰度掩模

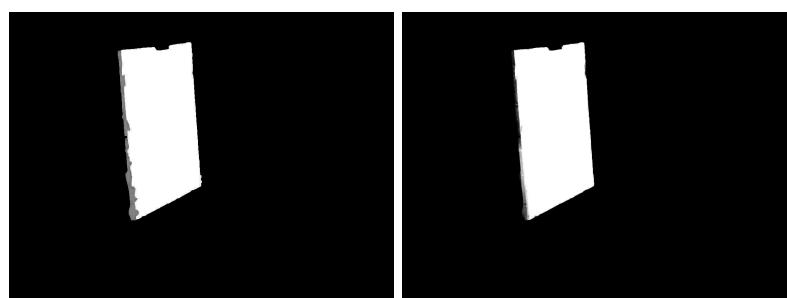


图 27: 模板 3 的 trimap 和灰度掩模

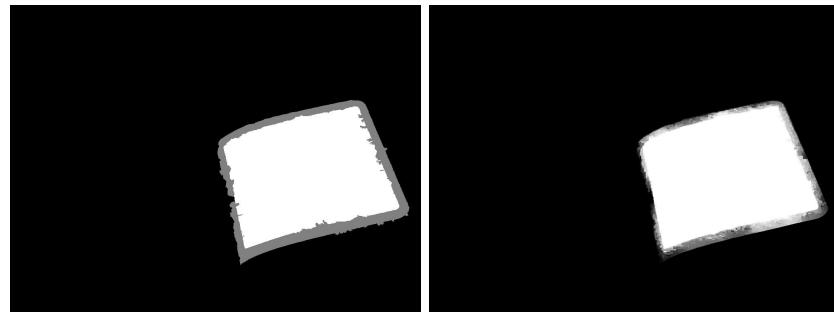


图 28: 模板 4 的 trimap 和灰度掩模

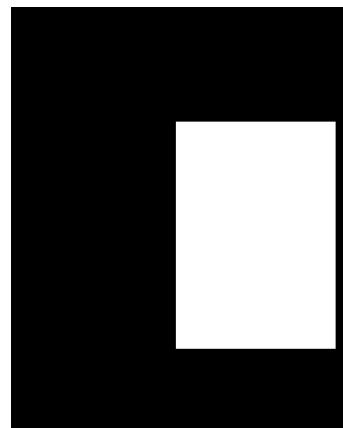


图 29: 模板 5 的掩模

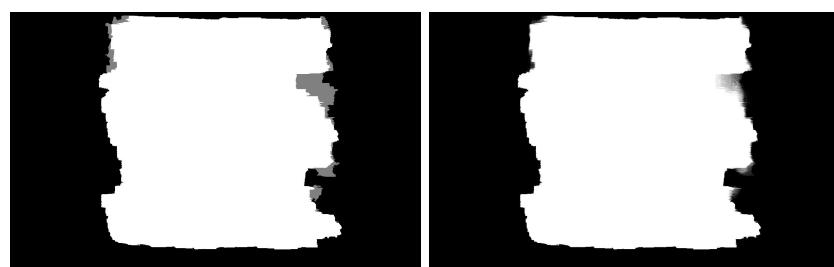


图 30: 模板 6 的 trimap 和灰度掩模

3.2 任务二

详细效果可以参阅生成的视频文件，其中 result_nohand.avi 是移除了手部遮挡的最终效果，result.avi 是不进行手部遮挡移除的效果，请按需食用。

参考文献

- [1] Zheng, Yuanjie, and Chandra Kambhamettu. "Learning based digital matting." Computer Vision, 2009 IEEE 12th International Conference on. IEEE, 2009.
- [2] Lu C, Xu L, Jia J. Combining sketch and tone for pencil drawing production[C],Proceedings of the Symposium on Non-Photorealistic Animation and Rendering. Eurographics Association, 2012: 65-73.