

综合作业 1：图像拼接

自 64 赵文亮 2016011452

1 导言

图像拼接是一项非常具有实用性的技术。例如，在室内空间狭小的条件下难以将物体拍摄完整，而通过拍摄多张照片并拼接的方式可以得到全景图像。本文实现了一套完整的图像拼接算法，该算法可以将随机顺序的图像拼接成全景图。该算法首先从所有输入图像中提取特征点，再根据这些特征点来确定互相匹配的图像。接着根据图像之间的匹配关系得到图像变换顺序和变换矩阵，最后将变换后的图像进行融合。

本文的后续主要内容如下：第 3 节介绍了图像特征点提取的 SIFT 算法，第 4 节介绍了图像匹配的 RANSAC 算法和检验判据，第 5 节基于最小生成树构建了所有图像的变换矩阵，第 6 节对变换后的图像进行融合，第 7 节展示了三组输入图像的拼接效果。

2 图像预处理

图像的预处理主要包括方向调整和大小调整两个方面。

- 由于输入图像内部存在方向信息，直接使用 MATLAB 读入时可能会出现旋转 90° 的情况，因此需要再正式读入图像前先作一定调整。
- 由于输入图像可能分辨率非常高，读入后直接开始拼接计算量较大，于是需要对图像进行适当的压缩。

3 图像关键点提取与匹配

本文采用了 [Lowe, 2004] 提出的 SIFT 算法进行关键点的提取¹。该算法主要分为以下几个步骤：尺度空间极值检测、关键点精确定位、关键点方向指定、关键点描述子生成。提取出关键点后，对描述子求解最近邻问题可以完成关键点的匹配。

3.1 尺度空间极值检测

设原始图像为 $I(x, y)$ ，将其变换到尺度空间为 $L(x, y, \sigma)$ ，则有

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \quad (3.1)$$

其中

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \quad (3.2)$$

σ 参数表述了高斯滤波器对图像的模糊程度，选择不同的 σ 值可以得到一系列尺度空间的图像。

¹ 使用开源库 VLFeat (<http://www.vlfeat.org/>) 完成关键点提取和匹配

使用 DoG (Difference-of-Gaussian) 可以实现对关键点的检测:

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \quad (3.3)$$

可见只需对得到的多个尺度空间的图像相减, 即可得到 $D(x, y, \sigma)$, 进而对其中的每一个点比较它的值与周围八个邻点的取值, 以及上下两层中共 18 个邻点的取值, 如果该点是最大或最小值点, 则保留; 否则舍弃该点。

3.2 关键点精确定位

对 $D(x, y, \sigma)$ 对某一采样点 (假设位于原点) 进行泰勒展开, 可得

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}} \mathbf{x} + \frac{1}{2} \mathbf{x}^T \frac{\partial^2 D}{\partial \mathbf{x}^2} \mathbf{x} \quad (3.4)$$

其中 $\mathbf{x} = (x, y, \sigma)$ 表示某一点相对于原点的偏移。式 (3.4) 对 \mathbf{x} 求导, 并令导数为 0, 可以得到极值点 $\hat{\mathbf{x}}$

$$\hat{\mathbf{x}} = -\frac{\partial^2 D^{-1}}{\partial \mathbf{x}^2} \frac{\partial D}{\partial \mathbf{x}} \quad (3.5)$$

从而可以计算出

$$D(\hat{\mathbf{x}}) = D + \frac{1}{2} \frac{\partial D^T}{\partial \mathbf{x}} \hat{\mathbf{x}} \quad (3.6)$$

接着将 $|D(\hat{\mathbf{x}})|$ 较小的点去除。进一步地, 对上述过程中求出的极值点进行进一步筛选, 去除对比度较低的点, 尤其是边缘点。计算每个点的 Hessian 矩阵

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{bmatrix} \quad (3.7)$$

进而通过判据

$$\frac{\text{Tr}(\mathbf{H})^2}{\text{Det}(\mathbf{H})} < \frac{(r+1)^2}{r} \quad (3.8)$$

可以去除边缘点。

3.3 关键点方向指定

得到关键点之后, 需要为它们指定方向。对于每一个关键点 $\hat{x}(x, y, \sigma)$, 从其对应的尺度图像 $L(x, y, \sigma)$ 中计算每一点的梯度的大小和方向:

$$\begin{aligned} m(x, y) &= \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2} \\ \theta(x, y) &= \tan^{-1} \left(\frac{L(x+1, y) - L(x-1, y)}{L(x, y+1) - L(x, y-1)} \right) \end{aligned} \quad (3.9)$$

为了求出关键点的方向, 将 $0 \sim 360^\circ$ 分成 36 个区间, 构建角度直方图。角度落在某个区间的点会以梯度大小和一个高斯权函数加权添加到直方图中, 这个高斯权函数以关键点坐标为中心, 关键点尺度的 1.5 倍为标准差。对得到的直方图进行抛物线插值, 可以求出极值对应的方向, 以此作为关键点的角度。另外, 直方图中所有高度大于峰值高度 80% 的方向也被认为是一个潜在的方向。

3.4 描述子生成

描述子的生成仍然需要在关键点的尺度参数对应的高斯模糊图像中，考虑关键点邻域的采样点的梯度。为了保证特征提取的旋转不变性，首先根据关键点的方向将描述子和梯度的坐标系旋转。接着将采样窗口以关键点为中心划分为 4×4 的区域，每个区域内部计算采样点的梯度方向的直方图，直方图的区间长度为 45° （共 8 个区间）。每个采样点首先经过一个中心在关键点、标准差为采样窗口边长一半的高斯函数加权，再按照与直方图区间中心的距离添加到直方图内。这样，对于每一个关键点可以得到一个 $4 \times 4 \times 8 = 128$ 维的描述子。最后为了去除光照的影响，对这个描述子进行归一化即可得到最终的描述子。

3.5 关键点匹配

关键点匹配本质上是寻找描述子的最近邻问题，但是精确求解的时间复杂度较高，尤其是在 128 维上，即便是最有效的算法（例如 k-d 树）也会消耗大量时间。另一种基于 k-d 树修改的算法 BBF（Best-Bin-First）可以在较短时间内以较高概率计算出估计的最近邻结果。

4 图像匹配

由于输入图像是乱序的，且可能存在干扰图像，我们首先需要计算图像之间的匹配关系，并且对于匹配的两张图像生成二者之间的变换矩阵。本文采用 RANSAC（Random sample consensus）[Fischler and Bolles, 1981] 得到可能存在的匹配及其变换矩阵，并对这些潜在的匹配通过概率模型进行进一步的校验。

RANSAC 是一种非常简单有效的算法。首先我们对任意两张图像进行关键点匹配，得到了一组关键点对。下面则随机从这组关键点对中抽取 4 对，并使用 DLT（direct linear transformation）算法 [Hartley and Zisserman, 2003] 计算变换矩阵。设每一对关键点中，源关键点的齐次坐标为 $\mathbf{x}_i = (x_i, y_i, 1)^T$ ，目的关键点的齐次坐标为 $\mathbf{x}'_i = (x'_i, y'_i, 1)^T$ ，我们要寻找的变换矩阵 H 需要满足

$$\mathbf{x}'_i = k \mathbf{H} \mathbf{x}_i, \quad i = 1, 2, 3, 4 \quad (4.1)$$

由于采用齐次坐标，常数 k 的存在并不影响变换后的结果。上式可以表示为 $\mathbf{x}'_i \times \mathbf{H} \mathbf{x}_i = 0$ ，设变换矩阵 \mathbf{H} 的每一行分别为 $\mathbf{h}_1^T, \mathbf{h}_2^T, \mathbf{h}_3^T$ ，则有

$$\begin{bmatrix} \mathbf{0}^T & -\mathbf{x}_i^T & y'_i \mathbf{x}_i^T \\ \mathbf{x}_i^T & \mathbf{0}^T & -x'_i \mathbf{x}_i^T \\ -y'_i \mathbf{x}_i^T & x'_i \mathbf{x}_i^T & \mathbf{0}^T \end{bmatrix} \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{bmatrix} = \mathbf{0} \quad (4.2)$$

令上式左边矩阵为 \mathbf{A}_i ，则 \mathbf{A}_i 可以简写为

$$\mathbf{A}_i = \mathbf{x}_i^T \otimes \begin{bmatrix} 0 & -1 & y'_i \\ 1 & 0 & -x'_i \\ -y'_i & x'_i & 0 \end{bmatrix} \quad (4.3)$$

其中 \otimes 符号表示 Kronecker 积。令

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \mathbf{A}_3 \\ \mathbf{A}_4 \end{bmatrix}, \quad \mathbf{h} = \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{bmatrix} \quad (4.4)$$

则有

$$\mathbf{A}\mathbf{h} = \mathbf{0} \quad (4.5)$$

求解式 (4.5) 即可得到转移矩阵。实际计算中，对 \mathbf{A} 做奇异值分解，得到 $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T$ ，取最小奇异值的特征向量即 \mathbf{V} 的最后一列作为 \mathbf{h} 。将得到的特征矩阵作用于所有的源关键点 \mathbf{x} 得到 $\tilde{\mathbf{x}}$ ，再将 $\tilde{\mathbf{x}}$ 与 \mathbf{x}' 中相距较近的点对的数目作为该次匹配的得分，即

$$score = \sum_i match(i), \quad match(i) = \begin{cases} 1, & \|\tilde{\mathbf{x}}_i - \mathbf{x}'_i\| < \epsilon \\ 0, & otherwise \end{cases} \quad (4.6)$$

其中 ϵ 为一个设定的阈值，本文中取 $\epsilon = 5$ 。接下来则重复上述从改组关键点对中采样、计算变换矩阵、计算匹配得分的过程，并将最终得分最高的以此采样得到的转移矩阵作为最终的转移矩阵。实验中发现，只要重复次数足够多，RANSAC 有很大概率找到正确的变换矩阵。

接着基于一个概率模型 [Brown and Lowe, 2007] 对 RANSAC 算法得到的匹配图像做进一步的验证。记两张图像 I_i 和 I_j 共有 n_f 对匹配特征点， I_j 经过变换后有 n_i 个点与 I_j 中对应点距离在阈值 ϵ 内，则如果有

$$n_i > \alpha + \beta n_f \quad (4.7)$$

则认为 I_i 和 I_j 两组图像确实存在匹配关系。其中 $\alpha = 8.0$ 和 $\beta = 0.3$ 为参数。最后，将比值 n_i/n_f 作为这两张图像归一化后的最终匹配得分。

5 图像变换

通过第 4 节的算法，我们已经得到了任何两张图像的匹配关系，对匹配的图像也得到了变换矩阵和匹配得分。本节将进一步确定拼接顺序，以及得到从原始图像到最终拼接后图像的变换矩阵。

5.1 变换树

将输入的原始图像作为节点，两张图像之间的匹配得分作为边的权重，可以构造一张变换图。图 1 展示了一张变换图的例子。输入图像为 $I_1 \sim I_6$ ，六张图像之间存在一些匹配关系，匹配得分即为节点之间边的权重。从图中可以看到，图像 I_1 与其它图像不存在匹配关系。

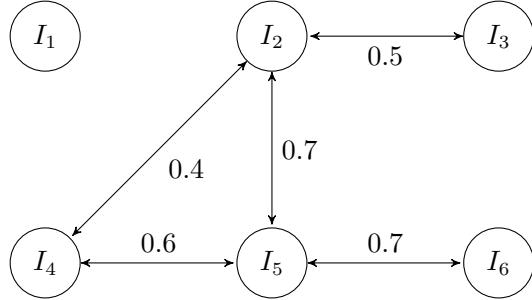


图 1: 变换图

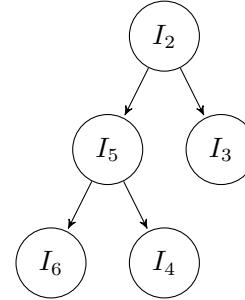


图 2: 变换树

为了确定参与拼接的图像以及拼接顺序，我们需要从变换图生成变换树。本文采用最小生成树的 Prim 算法 [Prim, 1957]，每次扩展树时找到匹配得分最高的节点（匹配得分越大，路径代价越小），得到 $I_1 \sim I_6$ 的变换树如图 2 所示。由于图像 I_1 不与其它图像匹配，它也不会出现在变换树中。

5.2 变换矩阵

求解每张图像到最后的拼接图像的变换矩阵需要两步：

- 基准图像估计
- 最终变换矩阵生成

首先，我们随机选择一个变换树中的节点作为根节点，初始化根节点的变换矩阵为单位阵，即

$$\mathbf{H}_{\text{root}} = \mathbf{I} \quad (5.1)$$

采用深度优先遍历的方式顺次生成变换矩阵，即

$$\mathbf{H}_i = \mathbf{H}_{\text{prt}(i)} \mathbf{H}_{i,\text{prt}(i)} \quad (5.2)$$

其中 $\text{prt}(i)$ 表示图像 i 在变换树中的父节点。于是图 2 中的搜索顺序为： $I_2 \rightarrow I_5 \rightarrow I_6 \rightarrow I_4 \rightarrow I_3$ 。

接下来计算每张图像经过变换后的边界范围，由于原始图像的边界经过透视变换后仍然是边界，所以这一步只需要对每张图像的四个顶点 $\mathbf{p}_{i1}, \mathbf{p}_{i2}, \mathbf{p}_{i3}, \mathbf{p}_{i4}$ 进行变换。则第 i 个图像的 x 边界范围是

$$[x_{\min,i}, x_{\max,i}] = \left[\min_{1 \leq k \leq 4} x_{\mathbf{p}_{ik}}, \max_{1 \leq k \leq 4} x_{\mathbf{p}_{ik}} \right] \quad (5.3)$$

同理可以计算出 y 方向的边界范围。令 $\bar{x}_i = (x_{\min,i}, x_{\max,i})/2$ ，再选择所有图像中 \bar{x} 位于中间的作为后续拼接的基准图像。

得到基准图像之后，以该基准图像作为根节点，再次遍历变换树，得到的一系列变换矩阵即为本文最终采用的变换矩阵。

5.3 图像插值

将原始图像 I_i 作用一个变换矩阵 \mathbf{H}_i 后，可以得到变换后各个像素点的坐标，但是在目标图像 I'_i 中仍有大量的像素点找不到原始图像中的对应点。本文使用双线性插值的方法解决这一问题。首先对目标图像 I'_i 中的坐标通过矩阵 \mathbf{H}'_i 反变换到 I_i 的坐标，不妨设某一点 $p(x', y')$ 反变换后变为 $p(x, y)$ ，且 (x, y) 落在由点 $p_{11}(x_1, y_1), p_{12}(x_1, y_2), p_{21}(x_2, y_1), p_{22}(x_2, y_2)$ 组成的方格中（包括边界），令

$$I'(x, y) = \frac{1}{(x_2 - x_1)(y_2 - y_1)} \begin{bmatrix} x_2 - x & x - x_1 \end{bmatrix} \begin{bmatrix} I(x_1, y_1) & I(x_1, y_2) \\ I(x_2, y_1) & I(x_2, y_2) \end{bmatrix} \begin{bmatrix} y_2 - y \\ y - y_1 \end{bmatrix} \quad (5.4)$$

即可完成插值。

图 3 展示了使用本节所述方法求取的最终变换矩阵，按照变换树深度优先顺序得到的前两张图像变换后的结果。

6 图像融合

在上一节中，我们已经得到了变换后的图像。直接将这些图像拼接起来，如图 4 所示。不难看出，虽然整体上来看图像中的关键点匹配正确，但是由于不同图像之间亮度的差异，在拼接处有明显的痕迹。本节将通过亮度调整和多频段融合来实现图像的融合。



(a) 第一张图像变换结果

(b) 第二张图像变换结果

图 3: 前两张图像变换结果



图 4: 未融合的拼接结果

6.1 亮度调整

对于任何两张重叠图像 I'_i 和 I'_j , 首先计算出它们的重合区域 O 内的亮度平均值。设它们对应的灰度图分别为 $Gray_i$ 和 $Gray_j$, 则

$$\begin{aligned} m_i &= \text{mean}_{p \in O} Gray_i(p) \\ m_j &= \text{mean}_{p \in O} Gray_j(p) \end{aligned} \quad (6.1)$$

以 I'_i 为基准来调整 I'_j 的亮度, 即令 $I'_j \leftarrow I'_j \cdot m_i/m_j$ 。实际拼接时, 也总是根据当前已拼接好的图像的亮度来调整新加入图像的亮度。

6.2 多频段融合

亮度调整使整体的色调较为一致, 但仍没有解决接缝处的过渡问题。本文使用多频段融合 [Burt and Adelson, 1983] 的算法, 实现了拼接的自然过渡。本文中实现的步骤为: 计算两张待融合图像的高斯金字塔, 进而计算它们的拉普拉斯金字塔; 基于 TPS 插值构造过渡函数将各层拉普拉斯金字塔融合; 从拉普拉斯金字塔重建出融合后的图像。

6.2.1 高斯金字塔与拉普拉斯金字塔

高斯金字塔是通过对图像不断下采样和高斯模糊构建的。设高斯金字塔 G 的层数为 $level$, 原图为 I 。生成高斯金字塔的过程如算法 1 所示。本文中该算法采用的高斯核大小为 5×5 , 标准差为 1, $reduce$ 函数以 2 像素的采样间隔对图像进行下采样。

算法 1: 高斯金字塔计算

```
Input : level, I
Output: G
G1 = I;
for i ← 2 to level do
    | Gi = reduce(Gi-1 * gaussian_kernel);
end
```

拉普拉斯金字塔则是通过将高斯金字塔逐层相减得到的, 具体过程如算法 3 所示。其中 $expand$ 函数将 G_{i+1} 扩大到和 G_i 同样大小。

算法 2: 拉普拉斯金字塔计算

```
Input : level, I
Output: L
for i ← 1 to level - 1 do
    | Li = Gi - expand(Gi+1, size(Gi));
end
Llevel = Glevel;
```

设待融合的两张图像为 I_A 和 I_B , 按照上述算法可以计算出它们的拉普拉斯金字塔分别为 LA 和 LB 。

6.2.2 TPS 过渡函数

得到拉普拉斯金字塔后，我们需要对两张图像的拉普拉斯金字塔的每一层图像进行融合。一种很直观的想法是构造函数 M ，令

$$LS_i = M \circ LA_i + (1 - M) \circ LB_i \quad (6.2)$$

其中 \circ 表示两个矩阵对应元素相乘。为了表述方便，设 $M_{A/B}, M_{B/A}, M_{A \cap B}, M_{A \cup B}$ 分别为“在图像 A 中但不在 B 中”、“在图像 B 中但不在 A 中”、“即在 A 中又在 B 中”、“在图像 A 或 B 中”的区域，如图 5 所示。

理想情况下， M 在 $M_{A/B}$ 中的部分应该为 1，在 $M_{B/A}$ 中的部分应该为 0，而在 $M_{A \cap B}$ 内应该均匀地从 1 过渡到 0。由于变换后图像形状不规则，直接构造过渡函数存在困难，本文使用 TPS (Thin plate splines) [Duchon, 1977] 求解了 M 在 $M_{A \cap B}$ 区域中的取值。首先获取 $M_{A \cap B}$ 与 $M_{A/B}$ 相邻的边界，这可以通过以下三步巧妙完成：

1. 对 $M_{A/B}$ 进行一次均值滤波，得到 $\hat{M}_{A/B}$

2. 构造矩阵 C ，

$$C(i, j) = \begin{cases} 1, & \hat{M}_{A/B}(i, j) > 0 \\ 0, & \text{otherwise.} \end{cases} \quad (6.3)$$

3. 边界 $Edge_1 = C \circ M_{A \cap B}$

同理可以提取出 $M_{A \cap B}$ 与 $M_{B/A}$ 相邻的边界。边界提取的结果如图 6a 和图 6b 所示。在 $Edge_1$ 上取一组采样点 $(x_i^{(1)}, y_i^{(1)}), 1 \leq i \leq N_1$ ，在 $Edge_2$ 上取一组采样点 $(x_j^{(2)}, y_j^{(2)}), 1 \leq j \leq N_2$ ，令

$$\begin{cases} M(x_i^{(1)}, y_i^{(1)}) = 1, & 1 \leq i \leq N_1 \\ M(x_j^{(2)}, y_j^{(2)}) = 0, & 1 \leq j \leq N_2 \end{cases} \quad (6.4)$$

使用式 (6.4) 中的所有采样点计算 TPS 插值函数，即可得到过渡函数 M 在 $M_{A \cap B}$ 中的范围。得到的 M 如图 6c 所示。

6.2.3 拉普拉斯金字塔融合与重建

有了过渡函数 M 后，我们就可以使用它将 LA 和 LB 的每一层融合得到新的拉普拉斯金字塔 LS ，并使用 LS 重建出融合后的图像。变换树深度优先遍历的第一张图片和第二张融合后的结果如图 7 所示，完全看

算法 3：拉普拉斯金字塔融合与重建

```

Input :  $LA, LB, M$ 
Output:  $S$ 
 $S \leftarrow 0;$ 
 $level \leftarrow length(LA);$ 
for  $i \leftarrow 1$  to  $level$  do
     $LS_i \leftarrow M \circ LA_i + (1 - M) \circ LB_i;$ 
     $M \leftarrow reduce(M * gaussian\_kernel);$ 
     $S \leftarrow S + expand(LS_i, size(S));$ 
end

```

不到拼接的痕迹。

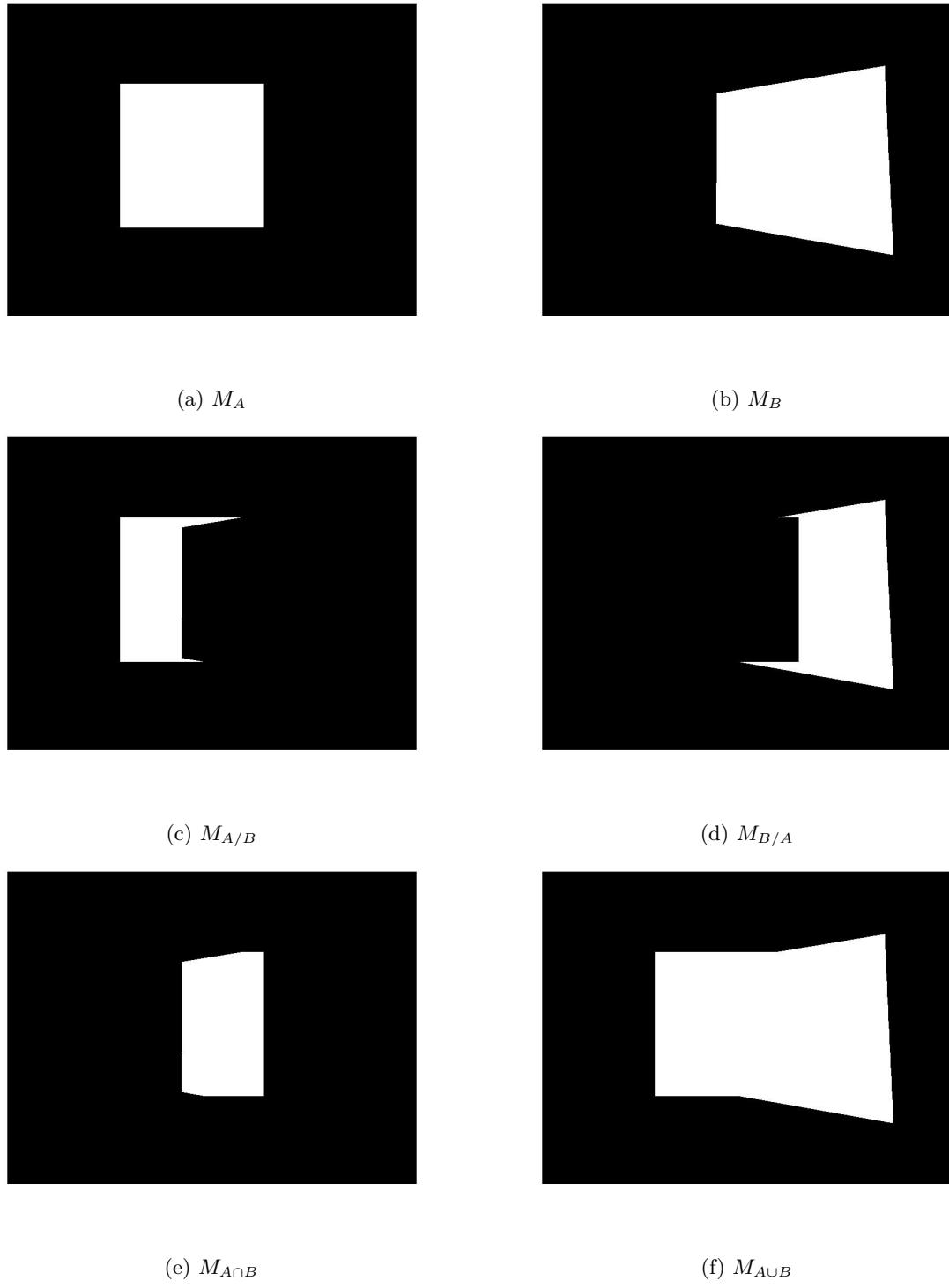


图 5: 待融合图像之间关系运算区域示意图

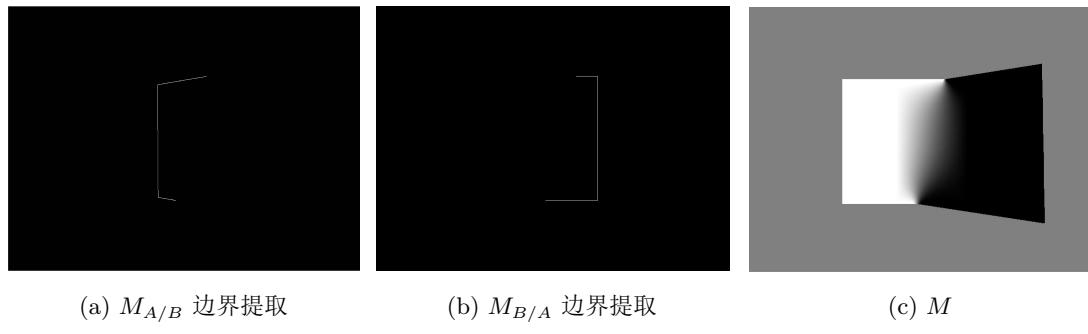


图 6: 过渡函数生成过程

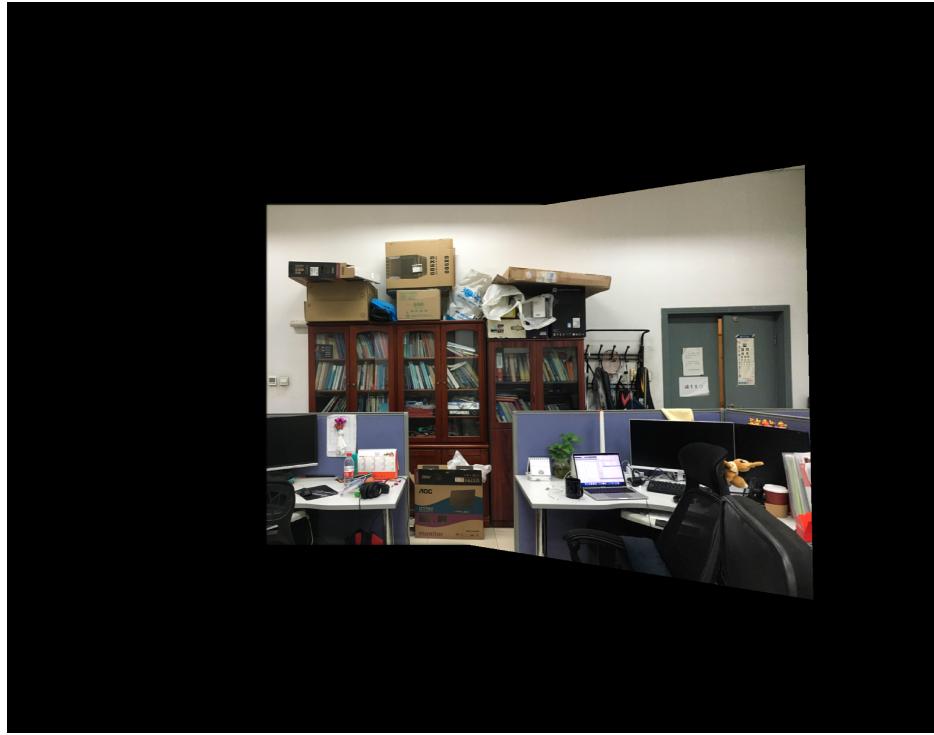


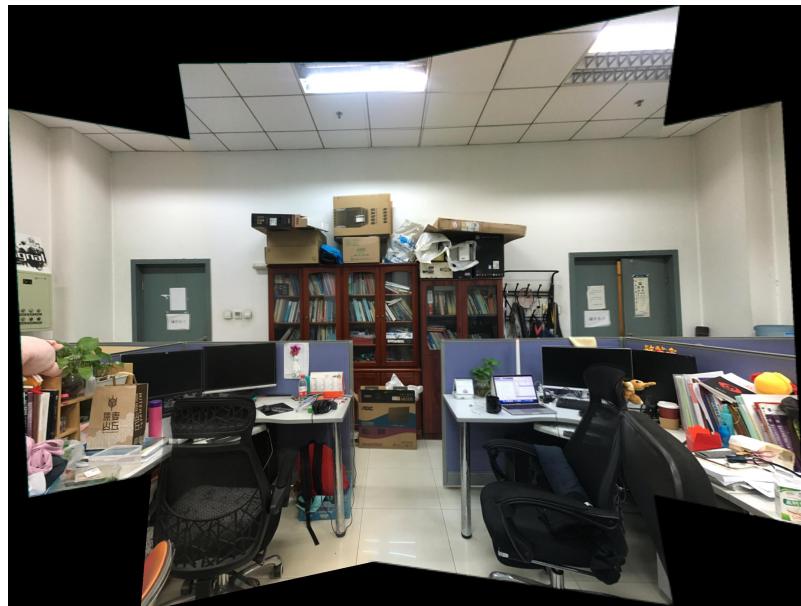
图 7: 前两张图像融合结果

7 拼接结果

至此，本文的图像拼接算法已经叙述完毕。下面展示在三组数据集（实验室、主楼、科协活动室）下的图像拼接效果，如图 8~10 所示。



(a) 原始图像



(b) 拼接图像



(c) 拼接图像（裁剪后）

图 8: 实验室拼接结果



(a) 原始图像



(b) 拼接图像

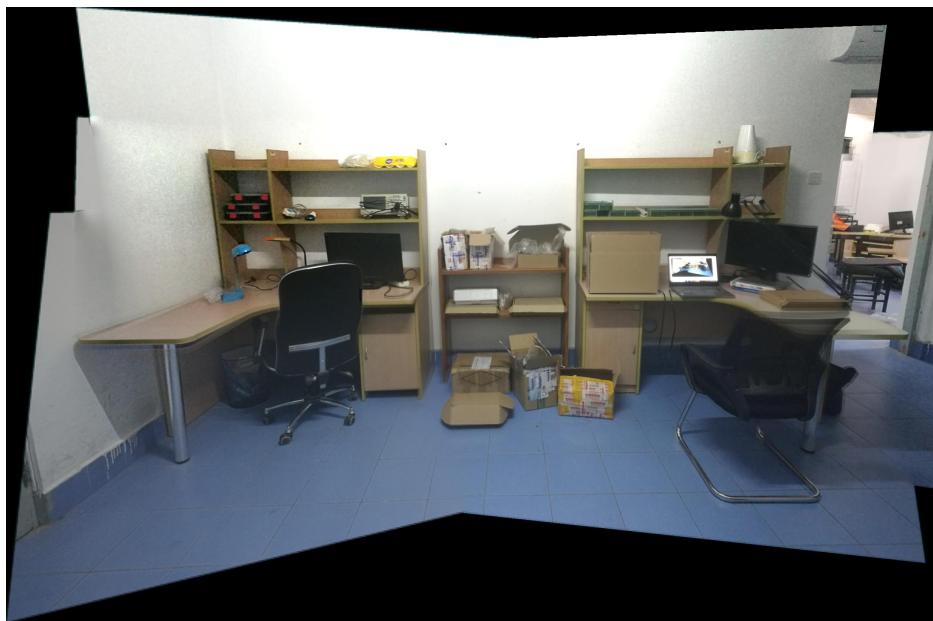


(c) 拼接图像 (裁剪后)

图 9: 主楼拼接结果



(a) 原始图像



(b) 拼接图像



(c) 拼接图像（裁剪后）

图 10: 科协活动室拼接结果

8 结语

本文实现了一套效果较好的图像拼接算法。该算法可以很好地处理输入图像乱序、存在无关图像等情况，并在整个拼接过程中不需要人为提供任何附加信息。但另一方面，由于该算法没有要求已知相机的焦距，在一些输入的情况下可能会导致拼接后的图像变形严重。如果在相机参数已知的条件下，可以利用相机参数对图像变换过程进行调整，从而得到鲁棒性更强的算法。

参考文献

- Brown, M. and Lowe, D. G. (2007). Automatic panoramic image stitching using invariant features. *International journal of computer vision*, 74(1):59–73.
- Burt, P. J. and Adelson, E. H. (1983). A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics (TOG)*, 2(4):217–236.
- Duchon, J. (1977). Splines minimizing rotation-invariant semi-norms in sobolev spaces. In *Constructive theory of functions of several variables*, pages 85–100. Springer.
- Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.
- Hartley, R. and Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge university press.
- Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.
- Prim, R. C. (1957). Shortest connection networks and some generalizations. *Bell system technical journal*, 36(6):1389–1401.