

这是标题

陈烁龙

2022 年 7 月 20 日

目录

1 角点细化	1
2 方向细化	1
3 棋盘角点探测的一般过程	1
3.1 计算 likelihood 响应	1
3.2 获取候选角	2
3.3 角点主方向	2
3.4 角点优化	2
3.5 棋盘生长	2

插图

1 原始影像	2
2 likelihood 影像	2
3 初步候选得到的角点	2
4 进一步筛选后的角点	2
5 候选角点的两个主方向	3
6 优化后的角点	3
7 优化后的两个主方向	3
8 恢复的棋盘结构 1	3
9 恢复的棋盘结构 2	3

表格

摘要

基于模板卷积和模态过滤，我们得到了候选角点（单位像素级）以及其得分和方向。接下来，我们需要进行亚像素级别的角点细化。

关键词：角点细化，高斯牛顿法

1 角点细化

对于一个真实的角度 c 而言，必然使得以下的误差函数趋于 0：

$$e_i(c) = g_{p_i}^T(c - p_i)$$

其中： $p_i = (x_i, y_i)^T$ 为角度 $c = (x_c, y_c)^T$ 邻域内的点， $g_{p_i} = (g_{x_i}, g_{y_i})^T$ 为像素点 p_i 的梯度向量。

对于棋盘格网点而言，如果 p_i 在格网点的黑白区域内部，由于其梯度接近 0，故使得 e_i 趋于 0；如果 p_i 在格网点的黑白区域交界处，由于其梯度垂直于向量 $c - p_i$ ，故也会使得 e_i 趋于 0。

综上，我们写出我们的目标函数：

$$f(c) = \min \sum \|e_i(c)\|^2$$

为求解该目标函数的最优解，我们对误差函数求解雅可比矩阵：

$$\begin{aligned} e_i(c) &= \begin{pmatrix} g_{x_i} & g_{y_i} \end{pmatrix} \begin{pmatrix} x_c - x_i \\ y_c - y_i \end{pmatrix} \\ &= g_{x_i}x_c + g_{y_i}y_c - (g_{x_i}x_i + g_{y_i}y_i) \\ &\rightarrow \frac{\partial e_i(c)}{\partial x_c} = g_{x_i} \\ &\rightarrow \frac{\partial e_i(c)}{\partial y_c} = g_{y_i} \\ &\rightarrow J_i = \begin{pmatrix} g_{x_i} \\ g_{y_i} \end{pmatrix} \end{aligned}$$

基于高斯牛顿法，我们有：

$$\begin{cases} H = \sum J_i J_i^T \\ g = -\sum J_i e_i \\ H \Delta X = g \end{cases}$$

迭代求解该方程并更新参数，我们即可得到对应的子像素级别的角点。注意：我们选取的领域是 11×11 。

2 方向细化

之前我们通过 mean shift 算法，找到了角点的两个模态。但是，当时我们只是简单的把它映射到 32 个 bin 的直方图中，不能精确的表示两个模态方向。为此我们需要对模态进行细化。

对于角度 c ，设其模态为 α_1, α_2 。二者的细化方法一致，我们选取第一模态进行讲解。对于模态 α_1 ，其对应的方向向量为 $v = (\cos \alpha_1, \sin \alpha_1)^T$ 。对于领域内满足以下条件的像素点 $p_i = (x_i, y_i)^T$ ：

$$|\cos(v, g_{p_i})| < 0.25$$

其中： g_{p_i} 为像素点 p_i 的梯度。换句话说，我们只考虑那些邻域内梯度和模态向量大概垂直的像素点。

对于满足条件的像素点，我们定义误差函数：

$$e_i = v^T g_{p_i} = \begin{pmatrix} g_x & g_y \end{pmatrix} \begin{pmatrix} v_x \\ v_y \end{pmatrix}$$

即有：

$$\rightarrow A_i X = 0$$

对于每一个符合条件的像素点，构建上述的方程。而后使用 SVD 分解法即可求解。

3 棋盘角点探测的一般过程

下图为原始影像。

3.1 计算 likelihood 响应

该步骤基于卷积的思想，通过构造的卷积核和图像做卷积操作，来计算不同像素对目标卷积

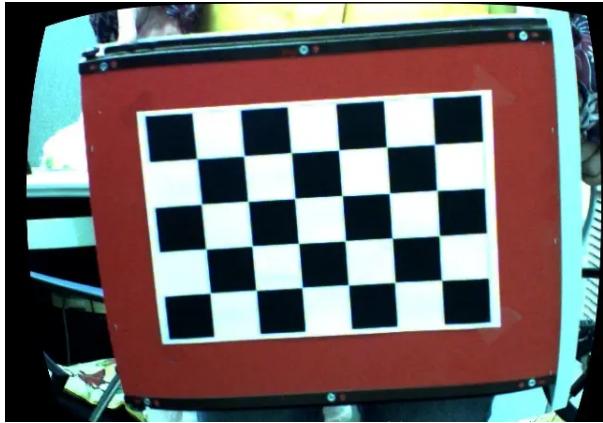


图 1: 原始影像

核的响应程度。对于角点而言，其响应值会相较于其他非棋盘角点而言，会大很多。



图 2: 计算 likelihood 响应: likelihood 影像

3.2 获取候选角

基于卷积操作，我们可以获得一些候选角点。当然在此之前需要进行非极大值抑制(NMS)。可以看到，里面是有一些错误的棋盘角点的，不过后续我们有操作将其去除。

3.3 角点主方向

我们统计每一个候选角点领域内的梯度方向，并使用 meanShift 算法，获取该角点的两个主方向(模态 mode)。另外，我们通过对两个主

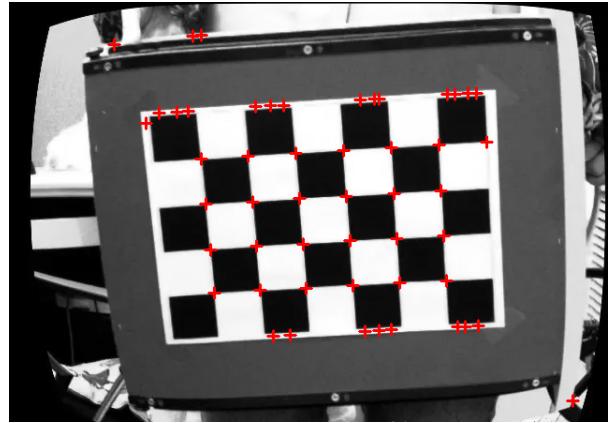


图 3: 获取候选角: 初步候选得到的角点

方向进行一定的限制，可以去除一些假的棋盘候选角点。

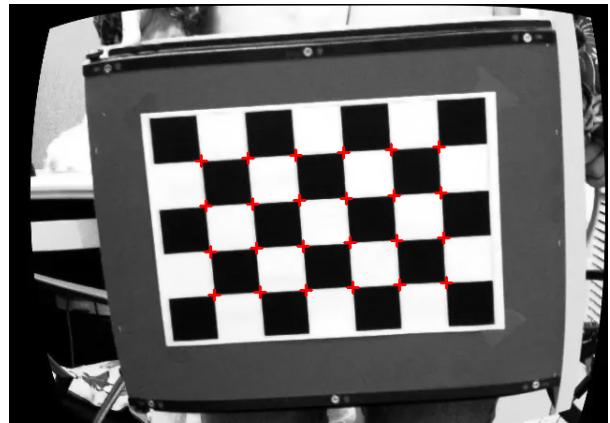


图 4: 角点主方向: 进一步筛选后的角点

3.4 角点优化

基于前文介绍的优化方法，我们将整像素级的角点优化到子像素级，另一方面，我们进一步优化棋盘格网点的两个主方向。

3.5 棋盘生长

最后，我们基于现有的角点，进行棋盘的结果恢复。

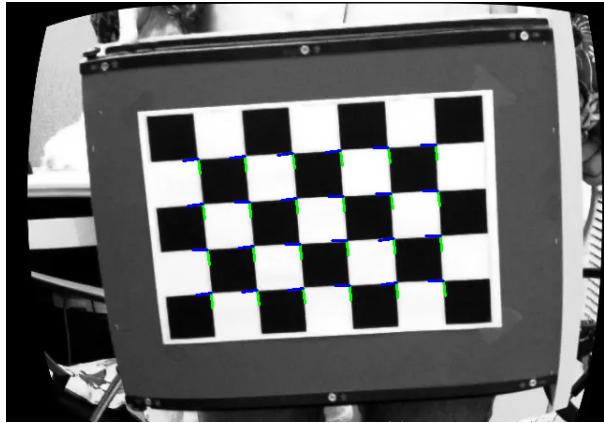


图 5: 角点主方向: 候选角点的两个主方向

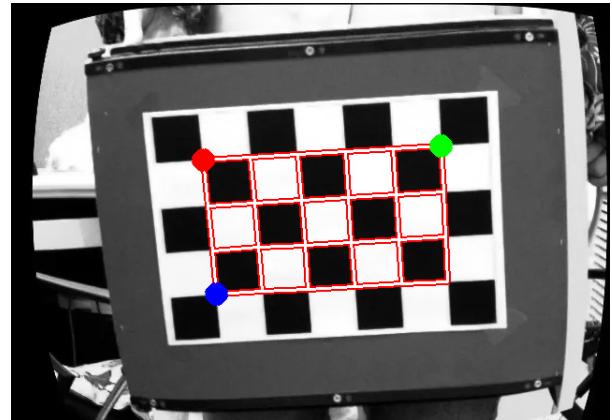


图 8: 棋盘生长: 恢复的棋盘结构 1

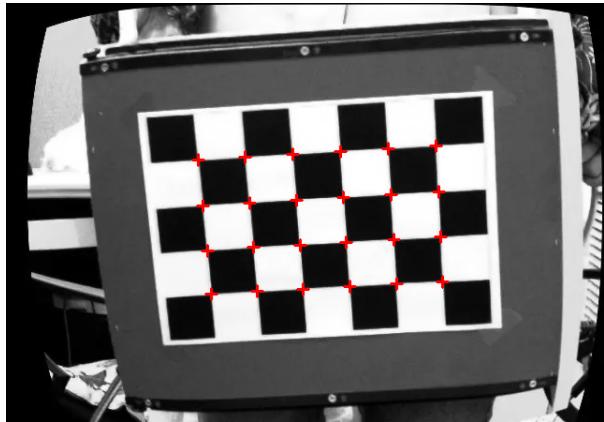


图 6: 角点优化: 优化后的角点

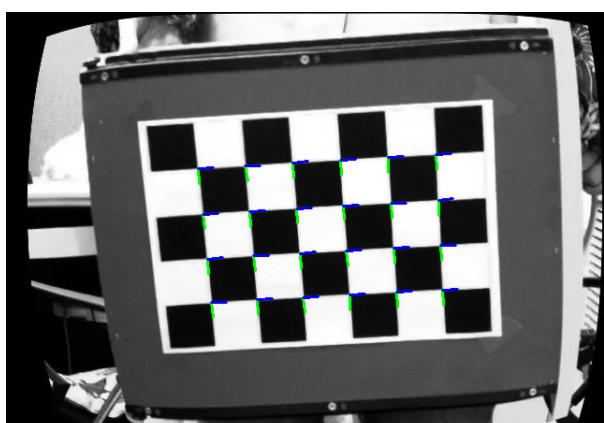


图 7: 角点优化: 优化后的两个主方向

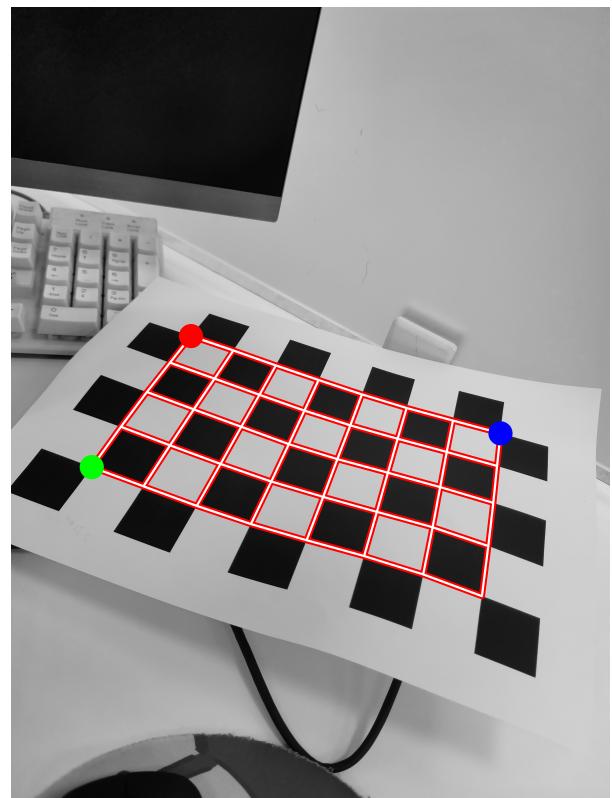


图 9: 棋盘生长: 恢复的棋盘结构 2