组 6-PJ1 相机标定小组报告

项目名称	PJ1 相机标定		
完成时间	2024/9/25	指导教师	路红
小组成员 （姓名学号）	专业学院		成员分工 （任务/在报告中的作用）
宋安洋 24210240291	计算机科学技术学院		完成了代码和实验、完成了报告第 1 部分、第 2.2 部分和第 3 部分
刘喆 24210240239	计算机科学技术学院		完成了报告第 2.1 部分
邓晨欣 24210240144	计算机科学技术学院		完成了报告第 2.3.1 部分
李天佑 24212010019	软件学院		完成了报告第 2.3.2 部分

复旦大学计算机科学技术学院

目　　录

摘要

在机器视觉领域，相机的标定是一个关键的环节，它决定了机器视觉系统能否有效的定位，能否有效的计算目标物。本小组按 PJ1 的任务要求，认真阅读了张正友的文章，尝试使用了 opencv 的 cv2 包分别选择了对第 1、2、4、5 个相机中每个相机拍摄的一组照片分别进行了相机标定，并完成了报告的撰写。

1　算法实现

1.1　算法流程

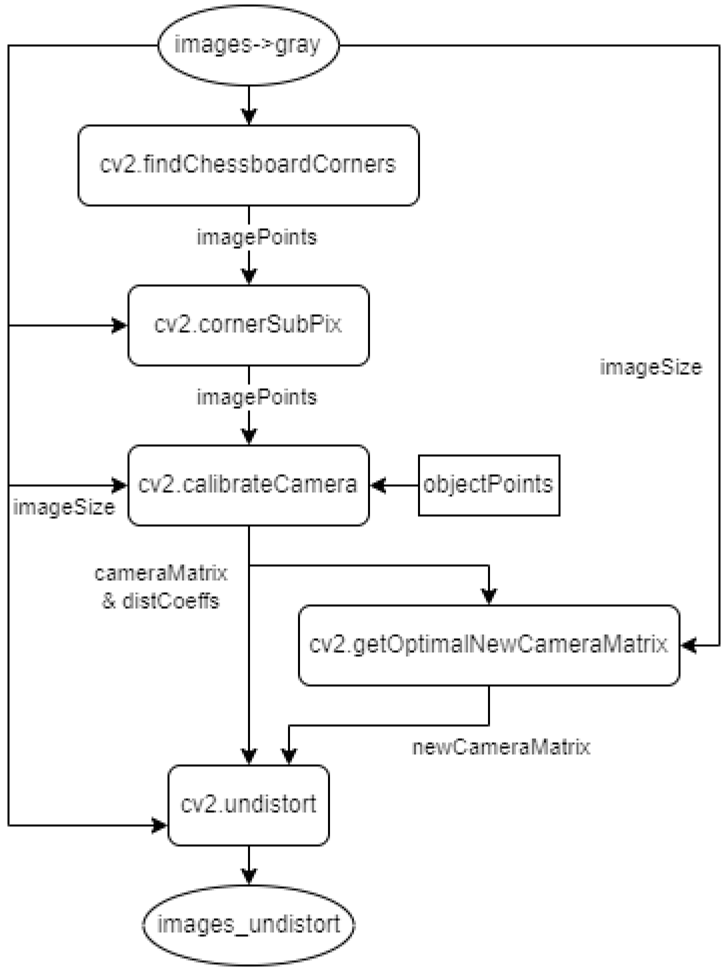

图1.1　算法流程图

对于输入的相机拍摄的一组标定板的图像，我们的最终目标是得到这组图像的单目标定的去畸变后的图像。我们可以用 cv2.undistort 函数去畸变，在这个过程中，我们需要提供 3*3 的相机内参矩阵 cameraMatrix、畸变参数 distCoeffs 和新的相机内参矩阵 newCameraMatrix。其中，newCameraMatrix 可以通过 cv2.getOptimalNewCameraMatrix 这个函数得到，同样的它也需要 cameraMatrix 和 distCoeffs。而这是我们可以通过相机标定来得到的，cv2.calibrateCamera 这个函数，而我们需要提供的是，世界坐标系中的 3D 坐标 objectPoints 和图像坐标系中的 2D 坐标 imagePoints。前者只是(x, y, z)，z=0，x, y 从 0 到 8 的一个 9*9 的排列组合（因为作业给的标定板是 10*10 的格子的），后者棋盘格的内角点，我们可以通过 cv2.findChessboardCorners 来寻找内角点得到。对于得到的这个内角点我们还可以通过 cv2.cornerSubPix 进一步优化一下，寻找亚像素角点。于是整个算法流程如图 1.1 所示。

1.2 数据分析

实际上我们认为该任务的数据也没有什么好分析的，可能最重要的是数一下标定板的格子数，然后每张图像用的都是 10*10 这样的一个标定板。然后其次，快速尝试一下寻找格角点，发现第三个相机拍的四张照片（19.jpg, 21.jpg, 22.jpg, 26.jpg） 可能因为拍得远了点，拍得不是很清楚，导致 corners 找不全，所以第三个相机的这四张图就不做了。然后对于每个相机所拍一组照片的像素大小的检测中，发现第一个相机拍的照片中，1.jpg 的像素大小是 4608*3456，这与该组其他照片的像素 1440*1080 不符，故剔除了 1.jpg。剩下的照片都是可以做的。

1.3 代码实现

我们封装了一个函数 calibrate，用来标定来自一个相机的一组照片。对第 1、2、4、5 个相机的照片组分别调用这个函数进行标定。这个函数的具体代码实现过程如下：首先对一组中的 n 个照片，我们需要提供一个长为 n 的列表 objectpoints，其中 n 个元素是重复的，是世界坐标系中的 3D 坐标，我们把这个 9*9=81 个(x,y,z)点构成的矩阵记作 objp。由于 z=0，np.zeros 初始化一个维度(81,3)的零矩阵，然后通过 np.meshgrid 得到 x,y 分别从 0 到 8 的一个 9*9 排列组合，objp 拼接 n 次就得到了 objectpoints。然后对每个图片矩阵 img = cv2.imread(f_name)，由于 opencv 的通道格式是 BGR，首先用 cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)得到对应的灰度图像 img_gray，然后用 cv2.findChessboardCorners(img_gray, (9, 9), None) 得到内角点 corners，然后用 cv2.cornerSubPix(img_gray, corners, (5, 5), (-1, -1), criteria)优化格角点，寻找亚像素格角点，得到 corners2。其中参数 winSize=(5,5)表示搜索窗口大小为（5*2+1）×（5*2+1）= 11×11。参数 zeroZone 表示死区的一半尺寸，即对多大的中央区不做计算，我们传入

(-1,-1)表示没有死区。对于参数 criteria 优化标准是，采用(cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 30, 0.001)，即同时按照停止算法迭代指定的精度是否达到 epsilon 以及迭代步长是否达到 max_iter 的标准，满足二者之一即停止迭代，max_iter 取的三十，精度 epsilon 取的 1e-3。然后对找内角点的结果，我们用 cv2.drawChessboardCorners(img, (9, 9), corners, True) 将其绘制了出来，其中 patternWasFound 参数指定为 True 表示内角点全部找到，则将各内角点依次连接起来。对这个相机拍的每张图片都寻找 corners2 并放入列表，最终得到图像坐标系中的 2D 坐标的一个列表 imagepoints。然后再求一个图像宽高大小 imageSize，就可以做相机标定了，ret, mtx, dist, rvecs, tvecs = cv2.calibrateCamera(obj_points, img_points, (w, h), None, None)，其中参数估计的内参矩阵 cameraMatrix 和估计的畸变系数 distCoeffs 都是所求的未知，直接传 None 即可。然后得到返回值 ret、cameraMatrix、distCoeffs、rvecs、tvecs，分别是重投影误差（误差越小标定结果越好）、估计的相机内参矩阵、估计的畸变系数、旋转向量的列表、平移向量的列表。接下来就可以取得新的相机内参矩阵 newCameraMatrix 和感兴趣区域 validPixROI 了，newcameramtx, roi = cv2.getOptimalNewCameraMatrix(mtx, dist, (w, h), 1, (w, h))，其中参数 imageSize 和 newImgSize 均为(w,h)，参数 alpha 控制缩放比例，按照通常值设为 1 即可。然后就可以对每个图像 dst = cv2.undistort(img, mtx, dist, None, newcameramtx)去畸变了，得到去畸变后的图像 dst，其中参数 R 是是双目相机的校准才要用到，直接传 None 默认为单位阵即可，P 就是传入新的相机内参矩阵来去畸变。对于之前 roi 感兴趣区域这个输出，它是由(x,y,w,h)构成，表示不受畸变影响的区域，dst1 = dst[y:y + h, x:x + w]也就是这个图像的最后结果了。

2　数学原理

2.1　Harris角点检测算法

cv2.findChessboardCorners() 函数在 OpenCV 中使用的是 Harris 角点检测算法的一个变种，用于检测棋盘格图像中的角点。这个函数的主要目的是找到棋盘格图像中内角点的坐标。

Harris 角点检测算法原理：Harris 角点检测算法是一种基于图像灰度变化的局部特征检测方法，用于寻找图像中的角点。算法基于以下原理：在角点附近，图像中的任意方向移动一个小的位移，像素灰度会有显著的变化。这就是说，角点附近的区域是具有高灰度变化的区域。

Harris 角点检测算法通过计算像素灰度值在不同方向上的变化率，来评估每个像素是否可能是角点。具体步骤如下：

首先计算移动窗口的灰度差值，记将图像窗口平移[u, v]，产生的灰度变化为：

$$\mathrm{E(u,v)} = \sum_{x,y} w(x,y)[I(x+u,y+v)-I(x,y)]^2 \tag{2.1}$$

其中，w(x,y)为窗口函数，为 0/1 函数或者高斯函数。I(x,y)为(x,y)位置的图像灰度。I(x+u,y+v)为(x+u,y+v)位置的图像灰度。为了减小计算量，利用泰勒级数进行简化公式：

$$I(x+u,y+v) = I(x,y) + I_x u + I_y v + O(u^2,v^2) \tag{2.2}$$

将公式(2.2)带入公式(2.1)得到：

$$E(u,v) = \sum_{x,y} w(x,y)[I_x u + I_y v + O(u^2,v^2)]^2 \tag{2.3}$$

其中，

$$\left[I_x u + I_y v\right]^2 = [u,v]\begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix} \tag{2.4}$$

所以对于局部微小的移动量[u,v]，可以近似得到下面的表达式：

$$E(u,v) = [u,v]M\begin{bmatrix} u \\ v \end{bmatrix} \tag{2.5}$$

$$M = \sum_{x,y} w(x,y)\begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \tag{2.6}$$

M 是 2*2 的偏导数矩阵，或者说梯度的协方差矩阵，可以进行对称矩阵的变化。如果利用两个特征值进行替代，则是通过判断两个特征值的大小来判定像素属性。具体来说，如果λ_1和λ_2都很小，图像窗口在所有方向上移动都无明显灰度变化，则是平坦区；如果λ_1和λ_2都较大且数值相当λ_1~λ_2，图像窗口在所有方向上移动都产生明显灰度变化，则是角点区；如果λ_1>>λ_2或者λ_1<<λ_2，则是边缘区。

在实际应用中为了能够应用更好的编程，定义了内角点响应函数 R，通过判定 R 大小来判断像素是否为角点。R 取决于 M 的特征值，其定义如下：

$$R = det(M) - k(trace(M))^2 \tag{2.6}$$

其中， $\mathrm{det(M)} = \lambda_1\lambda_2$， $\mathrm{trace(M)} = \lambda_1 + \lambda_2$ ，k 为一个经验常数(0.04-0.06)。定义 R>threshold 且为局部极大值点时为角点。具体来说，R 只与 M 的特征值有关，R 为小数值时，为平坦区；R 为大数值负数时，为边缘区；R 为大数值正数时，为角点区。

另外，Harris 角点检测算子对图像亮度和对比度具有部分不变性，且具有旋转不变性，但不具有尺度不变性。

cv2.findChessboardCorners() 函数利用了 Harris 角点检测的思想，在检测棋盘格图像中的内角点时，寻找局部区域的高变化区域，即可能的角点。这个函数会尝试在图像中找到所有的内角点，然后再进一步进行优化和筛选，得到精确的角点坐标。

2.2 亚像素角点的求法

本节中将解释 cv2.cornerSubPix()函数的数学原理

图像本来都是像素点，用整数表达坐标最自然，而亚像素内角点则是求出更加精确的小数坐标。设待求亚像素点的坐标为 q，设一个周围的点坐标为 p_i，(p_i-q)是我们得到的第一个向量，设 p_i 处灰度为 G_i 即第二个向量，那么 G_i • (p_i-q)一定=0，这是因为，假设第一个向量落在了格子内，那么梯度为 0，G_i=0；假设第一个向量在黑白格子相交的边界线上，梯度不为 0，但二者是垂直的。

对于公式：

$$G_i \cdot (p_i - q) = 0 \tag{2.7}$$

展开得：

$$G_i \cdot q = G_i \cdot p_i \tag{2.8}$$

最小二乘法求解：

$$G_i^T G_i q = G_i^T G_i p_i \tag{2.9}$$

即：

$$q = (G_i^T G_i)^{-1} \cdot (G_i^T G_i p_i) \tag{2.10}$$

当然，q 是要通过窗口内的一系列点 p_i 来求得的，对于各点处梯度应当求和，严格来说(2.10)的公式是不对的。另外，各点距中心距离不一，不可一视同仁，要引入权重比如高斯权重，设 p_i 的权重为 w_i，则最终公式为：

$$q = \sum_{i=0}^{N} (G_i^T G_i \omega_i)^{-1} * (G_i^T G_i \omega_i p_i) \tag{2.11}$$

对于迭代终止条件，一是可以指定最大迭代次数，二是精度即两次寻找的 $q_n - q_{n-1} \leqslant \varepsilon$，即认为是最优解。

2.3 张正友相机标定的解决方法

2.3.1 封闭解与最大似然估计

二维点用 $m=[u, v]^T$，三维点用 $M=[X, Y, Z]^T$ 表示。$\tilde{x}$表示增广向量，最后一个元素加 1:$\widetilde{m}=[u, v, 1]^T$，$\widetilde{M}=[X, Y, 1]^T$。建立一个常见的小孔成像模型：三维点 M 和它图像投影点 m 关系如下：

$$\mathrm{sm} = \mathrm{A}[\mathrm{R} \quad \mathrm{t}]\widetilde{M} \qquad \text{with}\mathbf{A} = \begin{bmatrix} \alpha & c & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.12}$$

式中 s 是任意的比例因子。（R, t）称为外参，R 是旋转矩阵，t 是平移矩阵。A 是相机内参矩阵，(uo, vo)是坐标的主点，α 和 β 是图像在 u 和 v 轴的比例因子，c 是描述两个坐标轴倾斜角的参数。我们简称 A^{-T} 为$(A^{-1})^T$或者$(A^T)^{-1}$。

一般情况下，我们假设模型平面在世界坐标系中的 Z 坐标为 0。R 的第 i 列旋转矩阵位 r。从公式(2.12)中可知：

$$s\begin{bmatrix}u\\v\\1\end{bmatrix}=\mathbf{A}[\mathbf{r}_1\quad\mathbf{r}_2\quad\mathbf{r}_3\quad\mathbf{t}]\begin{bmatrix}X\\Y\\0\\1\end{bmatrix}=\mathbf{A}[\mathbf{r}_1\quad\mathbf{r}_2\quad\mathbf{r}_3]\begin{bmatrix}X\\Y\\1\end{bmatrix} \tag{2.13}$$

我们仍然用 M 表示位于位图中的一点，当 Z 为 0 后，M 可以表示为 $M=[X, Y]^T$，同样地$\widetilde{M}=[X, Y, 1]^T$。因此点 M 和它的在图像上映射点 m 关系用单应矩阵 H 联系：

$$s\widetilde{\mathbf{m}}=\mathbf{H}\widetilde{M}\qquad \text{with}\mathbf{H}=\mathbf{A}[\mathbf{r}_1\quad\mathbf{r}_2\quad\mathbf{t}] \tag{2.14}$$

很显然，H 是一个 3×3 的系数矩阵。

对标定平面的一张图片，它的单应矩阵可以估计出来。令 $H=[h_1\ h_2\ h_3]$，由(2.14)式可知：

$$[\mathrm{h}_1\quad \mathrm{h}_2\quad \mathrm{h}_3]=\lambda \mathrm{A}[\mathrm{r}_1\quad \mathrm{r}_2\quad \mathrm{t}] \tag{2.15}$$

式中 λ 是任意的标量。由所学知识可知 r_1 和 r_2 是正交的，于是我们有

$$\mathbf{h}_1^T\mathbf{A}^{-T}\mathbf{A}^{-1}\mathbf{h}_2=0 \tag{2.16}$$

$$\mathbf{h}_1^T\mathbf{A}^{-T}\mathbf{A}^{-1}\mathbf{h}_1=\mathbf{h}_2^T\mathbf{A}^{-T}\mathbf{A}^{-1}\mathbf{h}_2 \tag{2.17}$$

对一个给定的单应性矩阵，对内参有两个基本的约束条件。因为这个矩阵有 8 个自由度和 6 个外参（3 个是旋转矩阵的和 3 个是平移向量的），这个条件下，我们只能得到两个内参的约束条件。

怎样有效地解决摄像机的标定问题呢。我们先给出一个封闭解，然后根据最大似然估计给出非线性的最优化解，最后再考虑透镜的径向畸变，给出解析解和非线性解。

令

$$\mathbf{B}=\mathbf{A}^{-T}\mathbf{A}^{-1}$$

$$\equiv\begin{bmatrix}B_{11}&B_{12}&B_{13}\\B_{21}&B_{22}&B_{23}\\B_{31}&B_{32}&B_{33}p\end{bmatrix}=\begin{bmatrix}\frac{1}{\alpha^2}&-\frac{c}{\alpha^2\beta}&\frac{cv_0-u_0\beta}{\alpha^2\beta}\\-\frac{c}{\alpha^2\beta}&\frac{c^2}{\alpha^2\beta^2}+\frac{1}{\beta^2}&-\frac{c(cv_0-u_0\beta)}{\alpha^2\beta^2}-\frac{v_0}{\beta^2}\\\frac{cv_0-u_0\beta}{\alpha^2\beta}&-\frac{c(cv_0-u_0\beta)}{\alpha^2\beta^2}-\frac{v_0}{\beta^2}&\frac{(cv_0-u_0\beta)^2}{\alpha^2\beta^2}+\frac{v_0^2}{\beta^2}+1\end{bmatrix} \tag{2.18}$$

注意 B 是对称的，定义一个六维向量

$$\mathbf{b}=[B_{11},B_{12},B_{22},B_{13},B_{23},B_{33}]^T \tag{2.19}$$

假设 H 的第 i 列向量为 h1=[h11, h2, h3]T，然后我们可以得到

$$\mathbf{h}_i^T=\mathbf{B}\mathbf{h}_j=\mathbf{v}_{ij}^T\mathbf{b} \tag{2.20}$$

式中，$\mathbf{v}_{ij}=[h_{i1}h_{j1},h_{i1}h_{j2}+h_{i2}h_{j1},h_{i2}h_{j2},h_{i3}h_{j1}+h_{i1}h_{j3},h_{i3}h_{j2}+h_{i2}h_{j3},h_{i3}h_{j3}]$于是，两个基本约束(2.16)(2.17)式可以写为齐次式：

$$\begin{bmatrix}\mathbf{v}_{12}^T\\(\mathbf{v}_{11}-\mathbf{v}_{22})^T\end{bmatrix} \tag{2.21}$$

如果观测了 n 张图片，迭代可以得到

$$\mathrm{Vb} = 0 \tag{2.22}$$

式中 V 是一个 2n*6 矩阵。若 n⩾3，通常会得到一个唯一解 b。若 n=2，令倾斜约束参数 c=0。即[0, 1, 0, 0, 0, 0]b=0，这作为额外的方程带入(2.22)式。(2.22)式的解是被熟知的与最小奇异解相关的特征向量 V^TVp。

b 一旦估计出来，我们可以计算内参矩阵 A。

求出 A 后，相应的外部参数也可以被计算出来，从(2.14)式中，可以得到：

$$\mathbf{r}_1 = \lambda\mathbf{A}^{-1}\mathbf{h}_1, \mathbf{r}_2 = \lambda\mathbf{A}^{-1}\mathbf{h}_2, \mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2, \mathbf{t} = \lambda\mathbf{A}^{-1}\mathbf{h}_3 \tag{2.23}$$

式中$\lambda = 1/\parallel \mathbf{A}^{-1}\mathbf{h}_1 \parallel = 1/\parallel \mathbf{A}^{-1}\mathbf{h}_2 \parallel$。当然，由于噪声数据，因此计算的矩阵 R= [$r_1$, r_2, pr_3]一般不能满足一个旋转矩阵的属性。

上面的解决方案通过最小化代数距离获得.一般情况下这是没有物理意义的.我们可以通过最大似然估计理论来完善。我们得到模型平面的 n 幅图片，模型平面上有 m 点.假设图像上像素点的噪声服从独立的同一分布.最大似然估计可以从通过求以下函数的最小值得到：

$$\sum_{i=1}^{n} \sum_{j=1}^{m} \parallel \mathbf{m}_{ij} - \widehat{\mathbf{m}}(\mathbf{A}, \mathbf{R}_i, \mathbf{t}_i, M_j) \parallel^2 \tag{2.24}$$

根据(2.14)式，上式中$\widetilde{\mathbf{m}}(\mathbf{A}, \mathbf{R}_i, \mathbf{t}_i, M_j)$是 M_j 点在第 i 幅图像上的投影。旋转矩阵 R 用三个参数的向量 r 表示。r 平行于旋转轴，而且大小等于旋转角度。R 和 r 关系符合罗逋里格斯公式同。求(10)式的最小值是一个非线性优化问题，可以通过 Lcvenberg-Marquardt 算法解决。它需要矩阵 A 的一个初始的猜测值，$\{\mathbf{R}_i, \mathbf{t}_i | i = 1 \ldots n\}$可使用在上一小节中描述的方法得到。

2.3.2　径向畸变的处理

到现在为止，我们没有考虑过摄像机的镜头崎变。然而，桌面相机的镜头通常会有显著的畸变，尤其是径向畸变。在本节中，我们只考虑前两个方面的径向畸变。根据过去工作的报告，镜头的畸变以径向分量为主，尤其是第一项占主导地位。同时过去工作也有发现任何更复杂的模型不仅没有帮助，而且会导致数值不稳定。

设(u, v)是理想（畸变可忽略）像素的图像坐标。$(\breve{u}, \breve{v})$对应的实际观测到的图像坐标。同样，(x, y)和$(\breve{x}, \breve{y})$的理想（无畸变）和实时（含有崎变）情况下的归一化图像坐标.我们有：

$$\breve{x} = x + x[k_1(x^2 + y^2) + k_2(x^2 + y^2)^2] \tag{2.25}$$

$$\breve{y} = y + y[k_1(x^2 + y^2) + k_2(x^2 + y^2)^2] \tag{2.26}$$

其中，k_1 和 k_2 均是径向畸变系数。径向畸变中心的主点是相同的。从$\tilde{u} = u_0 + \alpha\breve{x} + c\breve{y}$和$\breve{v} = v_0 + \beta\breve{y}$中可以得到：

$$\breve{u} = u + (u - u_0)[k_1(x^2 + y^2) + k_2(x^2 + y^2)^2] \tag{2.27}$$

$$\breve{v} = v + (v - v_0)[k_1(x^2 + y^2) + k_2(x^2 + y^2)^2] \tag{2.28}$$

交替径向畸变的估计：径向畸变一般很小，简单的忽略径向畸变后，可以用 3.2 中的方法估计另外的五个参数。一种策略是估计其他参数后再估计 k_1 和 k_2。然后从(2.27)和(2.28)中，我们对每幅图的每个点可以得到两个方程

$$\begin{bmatrix}(u-u_0)(x^2+y^2) & (u-u_0)(x^2+y^2)^2 \\ (v-v_0)(x^2+y^2) & (v-v_0)(x^2+y^2)^2\end{bmatrix}\begin{bmatrix}k_1 \\ k_2\end{bmatrix} = \begin{bmatrix}\breve{u}-u \\ \breve{v}-v\end{bmatrix} \tag{2.29}$$

n 幅图像中共有 m 个点，迭代所有方程得到一个 2mn 的方程组，或者以矩阵形式 Dk = d，其中 $k=[k_1, k_2]^T$，由最小二乘法得：

$$\mathbf{k} = (\mathbf{D}^T\mathbf{D})^{-1}\mathbf{D}^T\mathbf{d} \tag{2.30}$$

k_1 和 k_2 估计出来后，可以通过解(2.24)式来重新估计其他参数。其中可以用式(2.27)(2.28)来代替$\widehat{\mathbf{m}}(\mathbf{A}, \mathbf{R}_i, \mathbf{t}_i, M_j)$交替使用这两个步骤，宜到收敛为止。

完整的最大似然估计：实验中，我们发现上述收敛过程比较慢.通过最小化下列函数来估计参数的完整集：

$$\sum_{i=1}^{n} \sum_{j=1}^{m} \parallel \mathbf{m}_{ij} - \widetilde{\mathbf{m}}(\mathbf{A}, k_1, k_2, \mathbf{R}_i, \mathbf{t}_i, M_j) \parallel^2 \tag{2.31}$$

根据方程(2.14)(2.27)(2.28)，其中$\breve{\mathbf{m}}(\mathbf{A}, k_1, k_2, \mathbf{R}_i, \mathbf{t}_i, M_j)$是点 M_j 在图像 i 中的投影。这是一个非线性最小化问题，可以通过 Uvenberg-Marquardta 法解决.旋转矩阵仍然还可用一个 2.3.1 节中所示的向量 r 来表示。A 初始值的选取和$\{\mathbf{R}_i, \mathbf{t}_i | i = 1 \ldots n\}$可以采用 2.3.1 中描述的技术。$k_1$ 和 k_2 初始值的选取可采用上一段所述的方法，或着直接将它们设置为 0。

3 实验结果

3.1 单目图像

我们分别对第 1、2、4、5 个相机进行了实验（第一个相机去除了 1.jpg），这里我们以第二个相机的 11.jpg, 12.jpg, 18.jpg 为例进行展示实验结果。

内角点的绘制如图 3.1，图 3.2，图 3.3 所示：

图3.1 11.jpg 内角点

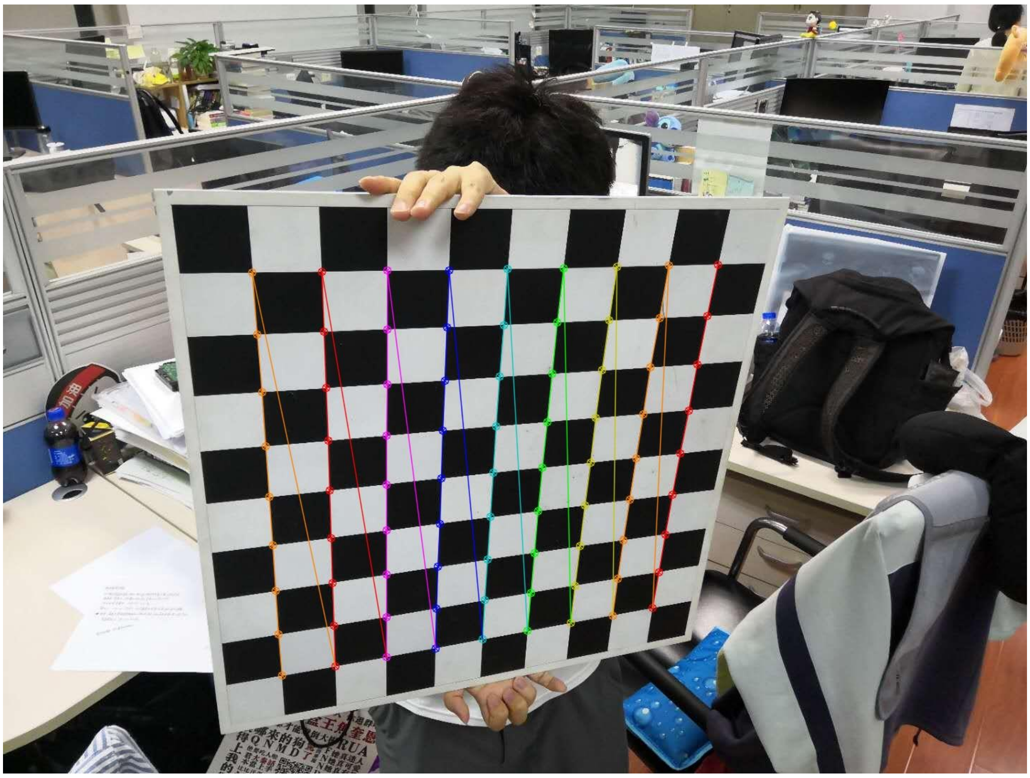

图3.2 12.jpg 内角点

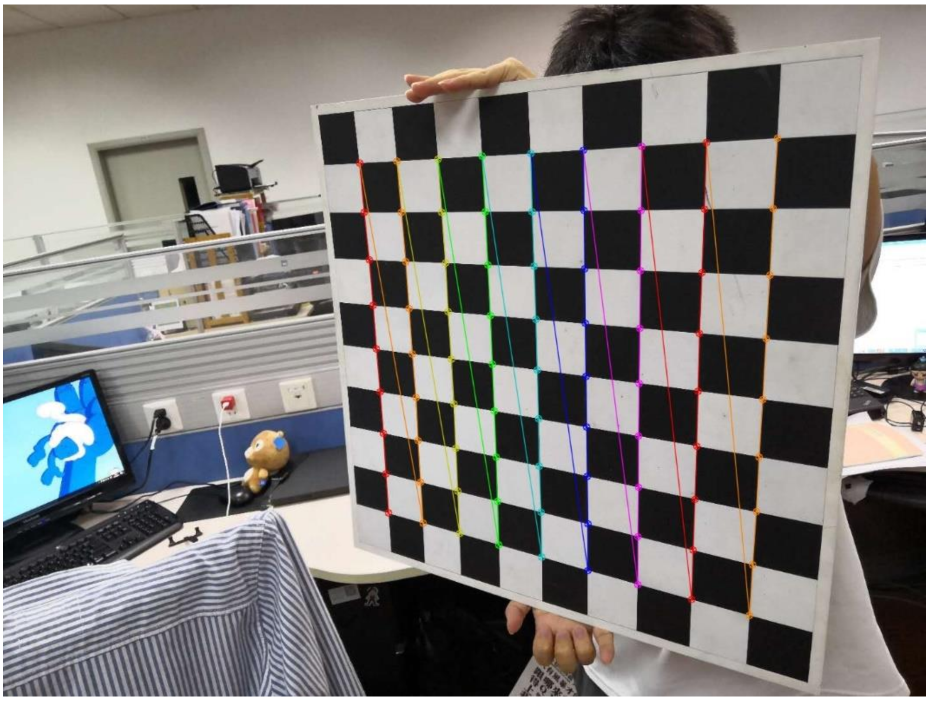

图3.3 18.jpg 内角点

相机标定的单目图像结果如图 3.4，图 3.5，图 3.6 所示：

图3.4 11.jpg 单目图像

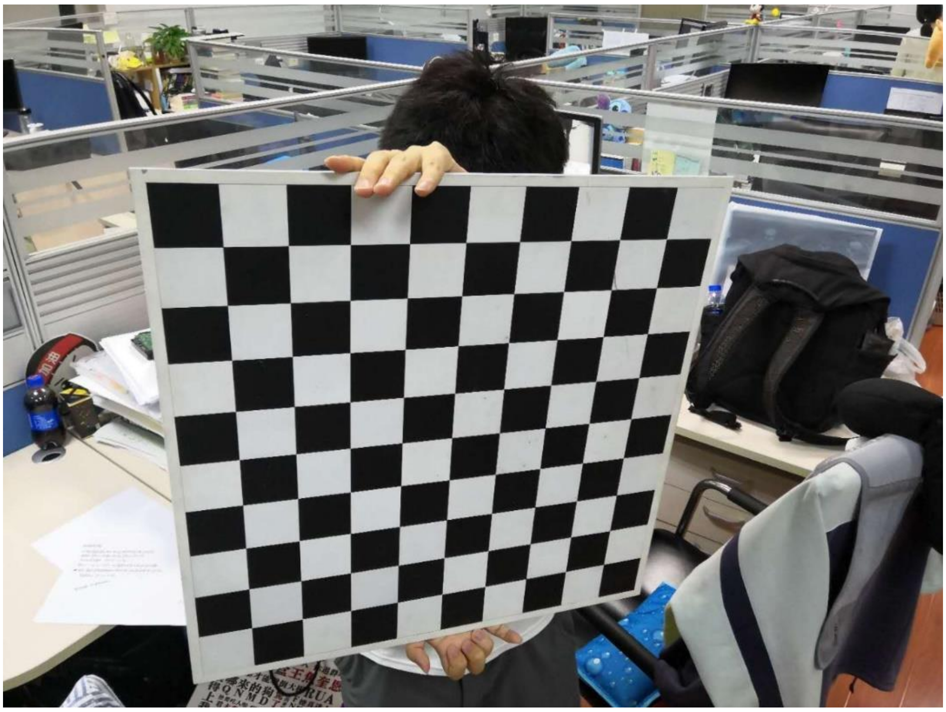

图3.5 12.jpg 单目图像

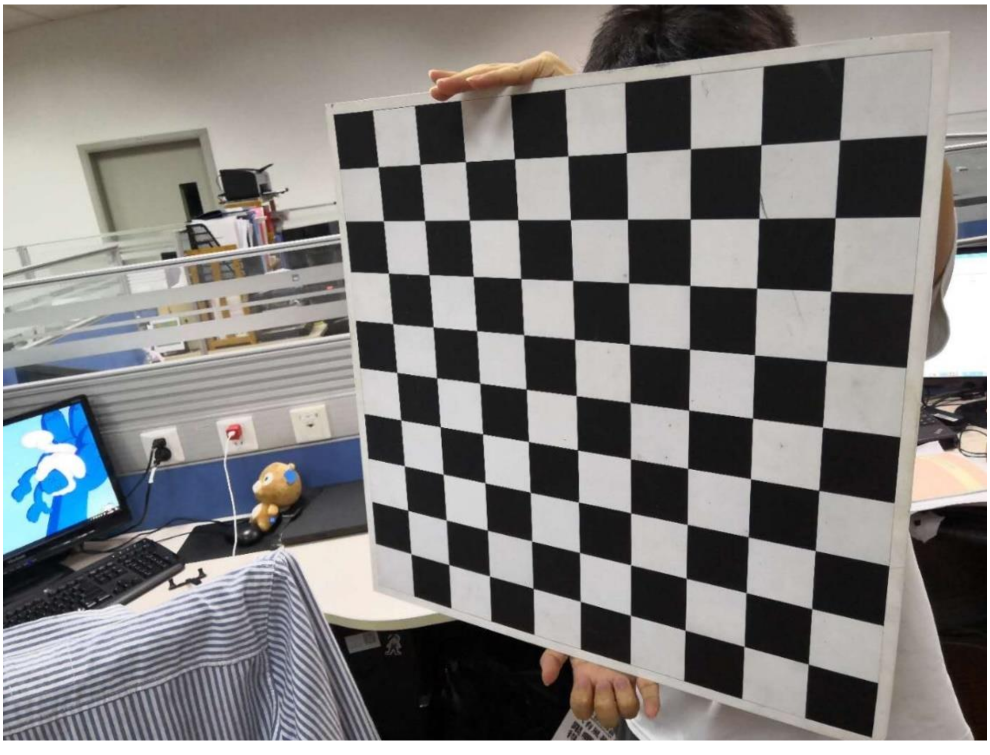

图3.6 18.jpg 单目图像

3.2 实验结果

以第二个相机为例，实验结果如下：

重投影误差 ret: 0.9066309021753808。该值越小，标定结果越好

估计的相机内参矩阵 cameraMatrix:

```
[[1.08076082e+03 0.00000000e+00 7.28826433e+02]
 [0.00000000e+00 1.07610418e+03 5.38418845e+02]
 [0.00000000e+00 0.00000000e+00 1.00000000e+00]]
```

估计的畸变系数 distCoeffs:

```
[[ 0.21440486 -0.82665858   0.00243381   0.00178348   0.96377005]]
```

旋转向量

```
rvecs: (array([[-0.33903519],
       [ 0.24753336],
       [ 1.42949237]]), array([[ 0.08657262],
       [-0.60492128],
       [ 1.53106548]]), array([[-0.2358022 ],
       [ 0.65379375],
       [-1.48816568]]))
```

平移向量

```
tvecs: (array([[ 3.37121331],
       [-6.10462468],
       [19.20635948]]), array([[ 3.55061605],
       [-2.27955653],
       [14.18986016]]), array([[-1.9005243 ],
       [ 3.85529338],
       [16.50571479]]))
```

新的相机内参矩阵 new_camera_mtx:

```
[[1.09794383e+03 0.00000000e+00 7.30922467e+02]
 [0.00000000e+00 1.09095998e+03 5.40645770e+02]
 [0.00000000e+00 0.00000000e+00 1.00000000e+00]]
```

感兴趣区域 roi: (26, 19, 1391, 1041)

参 考 文 献

[1] Zhang Z. Flexible camera calibration by viewing a plane from unknown orientations[C]//Proceedings of the seventh ieee international conference on computer vision. Ieee, 1999, 1: 666-673.
[2] Zhang Z. A flexible new technique for camera calibration[J]. IEEE Transactions on pattern analysis and machine intelligence, 2000, 22(11): 1330-1334.
[3] Harris C, Stephens M. A combined corner and edge detector[C]//Alvey vision conference. 1988, 15(50): 10-5244.