

Pie-Lab 2025年暑期培训

徐浩
Pie-Lab
北京理工大学计算机学院
北京市海淀区中关村南大街
526358612@qq.com

Abstract

本文是“实践IV：了解相机成像模型、三维重建流程”的实验报告。

1 数学原理

1.1 成像原理

小孔成像见图 1， p 点经过相机光心打到成像平面，得到成像点 p' 。

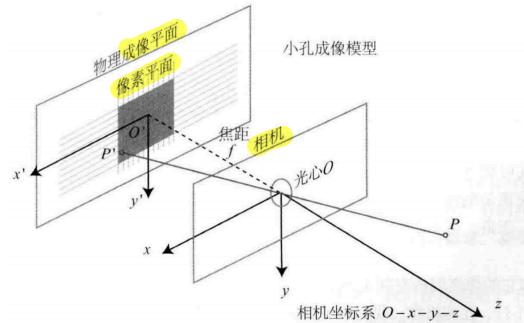


Figure 1: pinhole_imaging_model

自定义一个世界坐标（自定义原点、坐标轴、刻度和现实世界一致），设点 p 的世界坐标为 P_w 。

以相机的光心为原点，建立相机坐标系，则 p 还有一个相机坐标 P_c 。

首先要把 P_w 转为 P_c （欧式变换），需要坐标系的“旋转”加“平移”，数学原理如下。

设坐标系 1 的基为： $[e_1, e_2, e_3]$ ，向量 a 在坐标系 1 下的坐标为 $[a_1, a_2, a_3]^T$ ；设坐标系 2 的基为： $[e'_1, e'_2, e'_3]$ ，向量 a 在坐标系 2 下的坐标为 $[a'_1, a'_2, a'_3]^T$ ，则有：

$$[e_1, e_2, e_3] \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = [e'_1, e'_2, e'_3] \begin{bmatrix} a'_1 \\ a'_2 \\ a'_3 \end{bmatrix}$$

因此坐标系 2 到坐标系 1 的变换公式如下，其中 R 称作旋转矩阵：

$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} e_1^T e'_1 & e_1^T e'_2 & e_1^T e'_3 \\ e_2^T e'_1 & e_2^T e'_2 & e_2^T e'_3 \\ e_3^T e'_1 & e_3^T e'_2 & e_3^T e'_3 \end{bmatrix} \begin{bmatrix} a'_1 \\ a'_2 \\ a'_3 \end{bmatrix} \stackrel{\text{def}}{=} Ra'$$

然后加上一个平移向量 t 即可，因此同一个向量在两个坐标系中的坐标变换的完整公式如下：

$$a_1 = R_{12}a_2 + t_{12}$$

为了简化表示，引入齐次坐标， T 称为外参矩阵，包含了外参 R 和 t ：

$$\begin{bmatrix} a' \\ 1 \end{bmatrix} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ 1 \end{bmatrix} \stackrel{\text{def}}{=} T \begin{bmatrix} a \\ 1 \end{bmatrix}$$

所以 P_w 变换到 P_c 公式如下：

$$P_c = TP_w$$

p 打在成像平面上，得到成像点 p' ，则二者在相机坐标系下的坐标满足相似三角形，习惯性把 p' 放在 p 同侧，见图2。

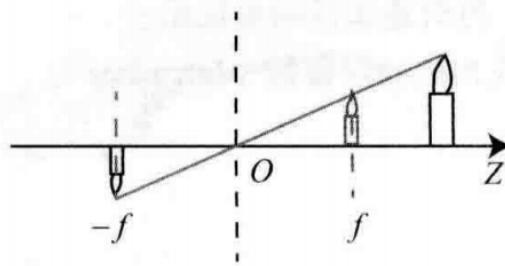


Figure 2: similar_triangle

因此 p 的相机坐标到 p' 的相机坐标：

$$\frac{z}{f} = \frac{x}{x'} = \frac{y}{y'}$$

在成像平面上还有一个像素平面，以图像的左上角为原点建立直角坐标系，则成像平面上的相机坐标系 o 、像素坐标系 o' 的关系见图3。

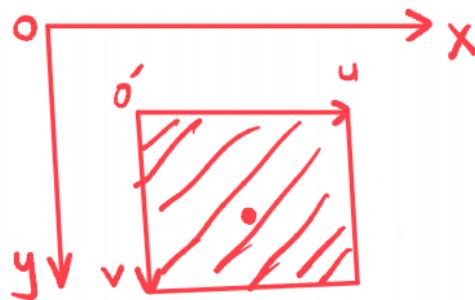


Figure 3: camera_pixel

p' 的相机坐标到像素坐标 P_p 的公式如下：

$$\begin{aligned} u &= \alpha x' + c_x \\ v &= \beta y' + c_y \\ z &= f \end{aligned}$$

又因为：

$$\begin{aligned} x' &= \frac{f}{z} x \\ y' &= \frac{f}{z} y \end{aligned}$$

代入像素坐标公式，并令： $\alpha f = f_x$ 、 $\beta f = f_y$ ，引入齐次坐标，得到 p 的相机坐标 P_c 到像素坐标 P_p 、世界坐标 P_w 的公式，其中 K 称为内参矩阵：

$$\begin{aligned} z \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} &= \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \\ zP_p &= KP_c = KTP_w \end{aligned}$$

1.2 标定原理

准备一张棋盘格，利用其中的交点进行标定。

首先写出交点的世界坐标。测量格子边长，以左上角为原点，x 轴自左而右，y 轴自上而下，z 轴垂直。进而可以写出交点的世界坐标，其中 z 坐标全 0。

然后写出交点的像素坐标 x_{ij} 。利用 cv2.findChessboardCorners 方法，可以自动提取灰度图中的交点。

初始化内参、外参。

利用上述世界坐标到像素坐标的公式，求出像素坐标 \hat{x}_{ij} ，然后求二者的误差如下，其中 N 张图像、每张图像 M 个交点。

$$\sum_{i=1}^N \sum_{j=1}^M (x_{ij} - \hat{x}_{ij})^2$$

然后更新内参、外参。

重复上述过程，不断优化，让误差减小，就可以得到较好的内参、外参，可以使用 cv2.calibrateCamera 自动完成。

1.3 重建原理

三维重建是从多张二维图像中恢复场景三维结构的技术，包括 SfM 和 MVS 两个核心步骤，前者解决相机姿态和场景稀疏结构问题，后者则在此基础上生成稠密三维点云。

1.3.1 SfM

SfM (Structure from Motion) 的基本流程如下：

1. 特征提取与匹配

- 对输入的每张图像提取稳定的局部特征（如SIFT、ORB等），这些特征包含位置、尺度、方向和描述子（用于匹配）。
- 在图像间进行特征匹配，找到不同视角下的对应点对（同一三维点在不同图像中的投影）。

2. 初始相机姿态估计（两视图重建）

- 从两张图像的对应点中，通过基础矩阵(Fundamental Matrix)或本质矩阵(Essential Matrix)估计相机间的相对姿态。
 - 基础矩阵 F : 描述两个图像平面上对应点的几何约束(与相机内参无关)，满足 $x_2^T F x_1 = 0$ (x_1, x_2 为对应点像素坐标)。
 - 本质矩阵 E : 当已知相机内参时， $E = K^T F K$ ，直接关联两个相机坐标系下的对应点($X_2^T E X_1 = 0$)，可分解得到旋转矩阵 R 和平移向量 t (尺度未知)。
- 通过三角化(Triangulation)计算对应点的三维坐标(以第一张相机为坐标系原点)。

3. 增量式重建（多视图扩展）

- 以初始两视图为基础，逐步加入新图像：
 - 对新图像，通过已有的三维点和图像特征匹配，估计其相对于已有相机的姿态(PnP问题: Perspective-n-Point)。
 - 用新相机的姿态三角化新的三维点，补充场景结构。
- 全局优化：由于增量式方法会累积误差，需通过光束平差法(Bundle Adjustment, BA)对所有相机姿态和三维点坐标进行联合优化，最小化重投影误差(三维点投影到图像上的坐标与实际特征点的偏差)。

SfM的输出结果如下：

- 所有图像的相机内外参(内参通过BA优化，外参即姿态)。
- 场景的稀疏三维点云(仅包含特征点，数量较少)。

1.3.2 MVS

MVS(Multi-View Stereo)的目标是在已知相机姿态的前提下，计算场景的稠密三维点云(覆盖场景表面的绝大多数点)。其核心是通过多张图像的像素级匹配，确定每个像素对应的三维深度。

MVS的基本原理如下：

1. 深度图估计(单视图深度预测)

- 对每张图像，为每个像素预测一个可能的深度范围(基于相机姿态和场景先验，如近大远小)。
- 通过平面扫描(Plane Sweeping)或光度一致性(Photometric Consistency)筛选最优深度：
 - 平面扫描：假设像素对应三维点在某一深度平面上，将其他图像的像素投影到当前图像，通过匹配度(如SSD、NCC)判断深度是否合理。
 - 光度一致性：同一三维点在不同图像中的像素颜色/亮度应相似(忽略光照变化和遮挡)。

2. 深度图融合与优化

- 每张图像会生成一张深度图(每个像素对应一个深度值)，但存在噪声和遮挡导致的错误。
- 利用相机姿态将所有深度图转换到统一的世界坐标系，通过一致性检查(如多个视图的深度是否接近)剔除错误点，融合得到稠密点云。
- 可选步骤：通过泊松重建、网格优化等生成三维网格模型。

MVS的输出结果如下：

- 场景的稠密三维点云(覆盖场景表面，精度取决于图像分辨率和视角数量)。

1.3.3 完整三维重建流程

输入图像→特征提取与匹配(SfM)→估计相机姿态与稀疏点云(SfM)→光束平差优化(SfM)→稠密深度图估计(MVS)→深度图融合→稠密点云/网格模型。

2 实验结果

2.1 标定结果

选择一个棋盘格，拍摄 7 张图像，正面 1 张，绕每个轴左右各旋转一次，共计 6 张。例如，将相机绕着 y 轴左旋，得到的棋盘格见图4。

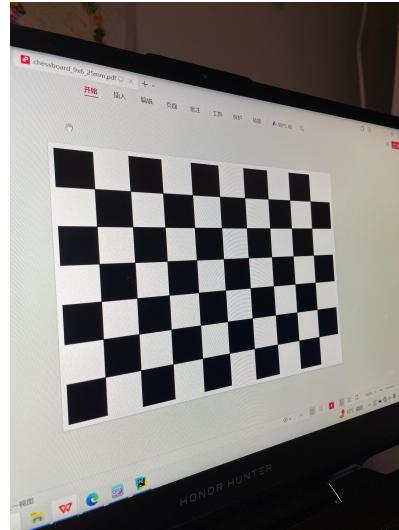


Figure 4: demo_img

标定结果如下，其中外参的旋转矩阵转为了欧拉角表示方法，便于理解：

内参矩阵 K

```
[3.10937744e+03 0.0000000e+00 1.51149970e+03]  
[0.0000000e+00 3.10850333e+03 2.03417107e+03]  
[0.0000000e+00 0.0000000e+00 1.0000000e+00]]
```

畸变系数

```
[ 1.55040621e-01 -5.69555550e-01 5.59448937e-04 8.50020005e-04 8.00733882e-01]
```

图像 1:

旋转向量: [-0.01208118 0.02375556 0.00315708]

平移向量 t: [-201.41267082 -248.02797239 483.41808235]

欧拉角(ZYX): Z=0.17°, Y=1.36°, X=-0.69°

图像 2:

旋转向量: [-0.05153448 0.03422195 -0.84146681]

平移向量 t: [-86.24529326 12.52713403 249.22130311]

欧拉角(ZYX): Z=-48.24°, Y=0.57°, X=-3.39°

图像 3:

旋转向量: [0.06456167 -0.02258622 0.87682857]

平移向量 t: [-10.25170183 -98.2082708 227.79266597]

欧拉角(ZYX): Z=50.19°, Y=-2.65°, X=2.71°

图像 4:

旋转向量: [-0.02870908 -0.55668183 -0.05640646]
 平移向量 t: [-51.34408697 -36.06344592 186.46605605]
 欧拉角(ZYX): Z=-3.09°, Y=-31.92°, X=-0.81°
 图像 5:
 旋转向量: [0.02143244 0.44653932 0.02917172]
 平移向量 t: [-76.5492159 -57.20112085 229.63332436]
 欧拉角(ZYX): Z=2.09°, Y=25.56°, X=1.72°
 图像 6:
 旋转向量: [-0.42000098 -0.02433906 -0.02992186]
 平移向量 t: [-74.2248263 -17.64964924 239.16165394]
 欧拉角(ZYX): Z=-1.38°, Y=-1.71°, X=-24.05°
 图像 7:
 旋转向量: [0.57991595 0.05742308 -0.01083493]
 平移向量 t: [-74.7341589 -84.70748614 209.38520934]
 欧拉角(ZYX): Z=0.34°, Y=3.28°, X=33.25°
 平均重投影误差:
 0.1041663605925199

2.2 重建结果

实验中发现，房间的重建结果很不好，而且非常耗时。
 因此选择了一个奖杯来重建(考虑到它兼有棱柱、曲面、文字等丰富元素)，对其拍摄了 30 张图像，稀疏重建结果见图5。
 稠密重建运行了一个多小时还是没有得到结果。



Figure 5: sparse_reconstruction

3 疑惑

本实验存在的疑惑如下：

- **稠密重建** 实验中发现，利用 colmap 进行重建时，只有 30 张图像，去重建一个奖杯：稀疏重建很快就能完成；稠密重建一直跑不出结果，利用 1660ti 跑了一个多小时，还在运行中。我想问问这是正常现象吗？