相机棋盘格标定 python-cv2 的实现

Dezeming Family

2023 年 5 月 4 日

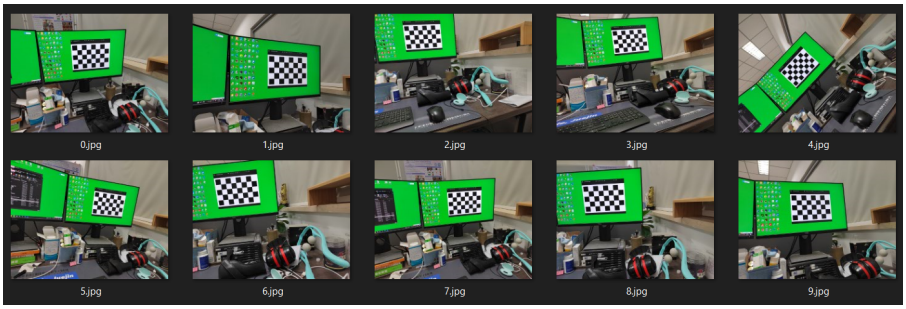
DezemingFamily 系列文章和电子书**全部都有免费公开的电子版**，可以很方便地进行修改和重新发布。如果您获得了 DezemingFamily 的系列电子书，可以从我们的网站 [https://dezeming.top/] 找到最新的版本。对文章的内容建议和出现的错误也欢迎在网站留言。

目录

一 marker 的提取于相机标定

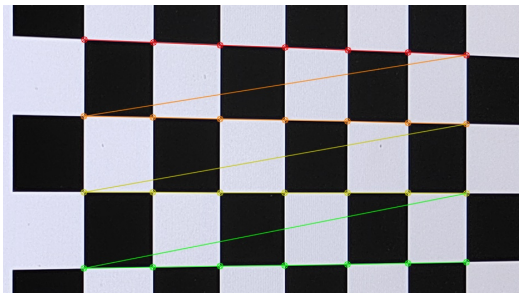
源码目录见 [3]。本节代码见 calibration.py 文件。

我们提供了一个棋盘格 marker，大家可以把 marker 打印或者直接在计算机上拍一些照片：

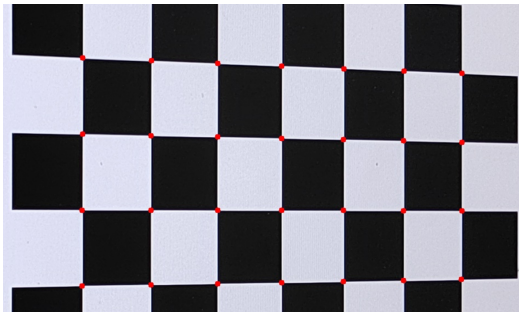


关于棋盘格，为了保持旋转不变，行数和列数必须一个是奇数一个是偶数，不能同时是奇数或者同时是偶数 [1]，否则则存在 180 度旋转的歧义。对于单台相机的校准不存在该问题，但如果相同的点需要由两个或更多的相机识别 (立体校准)，这种模糊性 (歧义) 必须不存在。

在《相机标定入门详解》中，我们已经介绍了棋盘格提取的相关函数和方法，我们的代码分为了多个部分，第一部分是“marker 的提取”，运行以后可以将 marker 角点标注出来：



在《相机标定入门详解》中，我们已经介绍了基本的标定方法，所以也不再赘述，重投影以后得到结果（选择其中一张图，此图中相机的外参平移和旋转都是已经求解得到的，我们将角点的世界空间坐标重投影到图像上）：



可以看到投影是比较精准的。

参数我们保存在了 camera_intr_opt.json 文件下。

二　参数的重新读取与投影

本节代码见 calibration2.py 文件。

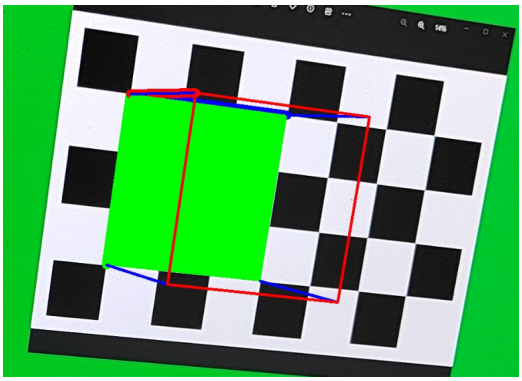
在进行投影之前，我们必须先检测拍摄到该图像时相机的位姿，这需要先检测到棋盘格的角点，方法和之前一样。

然后使用 solvePnP() 函数求解相机位姿，输入图像中点在世界坐标系的位置，以及这些点对应于图像中的坐标，然后求解得到相机的平移和旋转向量。

```
# 计算旋转和平移矩阵
retval, rvecs, tvecs, inliers = cv2.solvePnPRansac(objp, corners2, intrinsic, distortion)
```

投影的过程也较为简单,就是将世界坐标系下的物体投影到相机坐标系的图像空间,根据 cv2.projectPoints 函数将点投影到图像上，由于直线具有投影不变性（直线投影还是直线），所以把代表线段的两个端点投影到图像上再连接，就可以得到投影的直线了。

投影立方体结果如下：



参考文献

[1] https://zhuanlan.zhihu.com/p/353316030

[2] https://zhuanlan.zhihu.com/p/94244568

[3] https://github.com/feimos32/ComputerVision-Code-Implementation-and-Collection

[4] OpenCV-Python-Tutorial: OpenCV 官方教程中文版