1.实验名称及目的

1.1 实验名称

相机标定实验

1.2 实验目的

图像测量过程以及计算器视觉中，为确定空间物体某点的三维几何关系位置与其在图像中对应点之间的相互关系，必须建立相机成像的几何模型，模型的参数就是相机的参数。

1.3 关键知识点

本实验主要是实现通过Python接口VisionCaptureApi.py（见RflySimAPIs\RflySimSDK\vision目录）获取RflySim3D图像并实时更新相机参数（姿态、位置、FOV等）并进行相机标定实验。将传感器的图像通过死循环显示在屏幕上，并保存在一个以当前日期和时间命名的文件夹中，最后通过MATLAB进行标定。关键代码解析如下：

1）视觉接口使用

```
vis = VisionCaptureApi.VisionCaptureApi() \# 创建一个视觉传感器实例

vis.jsonLoad() \# 加载Config.json中的传感器配置文件

isSuss = vis.sendReqToUE4() \# 向RflySim3D发送取图请求

vis.startImgCap() \# 开启取图

vis.hasData[i] \# 图片i数据是否更新

pic1=cv2.cvtColor(img1, cv2.COLOR_BGRA2GRAY) \# 将图片转变为灰度图像

cv2.imshow(cv2.imshow("pic1", pic1)) \# 显示图片pic1图像

cv2.imwrite(os.path.join(path_img1, "{}.jpg".format(cnt)), pic1) \# 将当前图像pic1保存到

gray = cv.cvtColor(image, cv.COLOR_BGR2GRAY) \# 将输入的彩色图像转换为灰度图像

gray = cv.GaussianBlur(gray, (5, 5), 0) \#对灰度图像进行高斯模糊处理，以平滑图像并减少噪声

edged = cv.Canny(gray, 35, 125) # 使用 Canny 边缘检测算法检测图像边缘

(cnts, \_) = cv.findContours(edged.copy(), cv.RETR_LIST, cv.CHAIN_APPROX_SIMPLE)# 在

focallength_to_camera(knownWidth, knownLength, perWidth) #根据已知的物体宽度、长度和图像中

cv.drawContours(img1, [box1], -1, (0, 255, 0), 2) # 在图像 img1 上绘制 box1所表示的矩形轮
```

2）相机数量和参数配置

其中，视觉传感器的初始状态由本文件夹下的Config.json决定，主要包含以下配置项：

```
"SeqID":0：使用自动更新ID的方式，创建了SeqID为0的视觉传感器

"TypeID":1：传感器类型为RGB彩色图像

"TargetCopter":1：相机绑定在1号飞机上

"SendProtocol":[1,0,0,0,0,0,0,0]：传输模式1：UDP网络传输模式（图片使用jpeg压缩，点云直传）

"SensorPosXYZ":[0.3,0,0]：相机分布。
```

5）UE控制

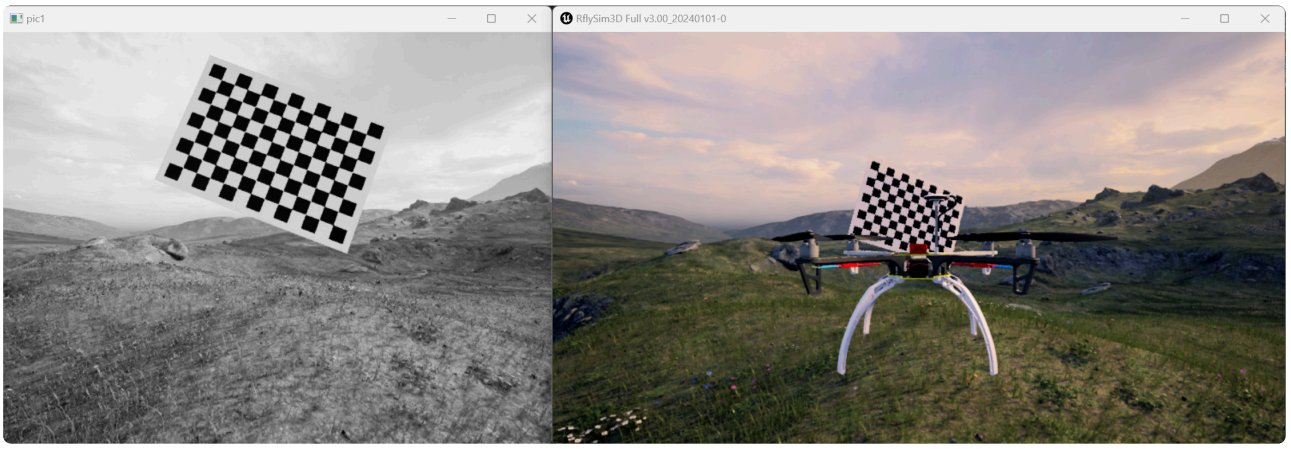
接口详细使用方法见： UE4CtrlAPI.py

```
ue = UE4CtrlAPI.UE4CtrlAPI() # 创建UE控制实例

ue.sendUE4Cmd('RflyChangeMapbyName GrassLands') #更新地图场景，类型为GrassLands

ue.sendUE4Cmd('r.setres 1280x720w',0) #发送指令，设置UE4窗口分辨率，注意本窗口仅限于显示，取图

ue.sendUE4Cmd('t.MaxFPS 30',0) #发送指令，设置UE4最大刷新频率30Hz，同时也是取图频率

ue.sendUE4Pos(1,0,0,[0,0,-8.086-1.5],[0,0,0]) #创建物体，id号为1，类型为0，速度为0，位置为[
```

5）其余代码说明

```
os.makedirs(path_dir) #新建一个文件夹，path_dir里面有文件夹路径，文件夹名字信息

timeInterval = 0.1\# 以10hz的频率进行控制

lastTime = lastTime + timeInterval \# 设置每一帧的处理结束时间

sleepTime = lastTime - time.time() \#计算休息时间，从而保持按照设定的频率执行代码

targetPosE=targetPosE+Error2UE4Map[j] \# 设置飞机位置

KNOWN_LENGTH = np.linalg.norm(a) \# 计算向量的欧几里得范数
```

2.实验效果



3.文件目录

例程目录：

[安装目录]\RflySimAPIs\8.RflySimVision\0.ApiExps\3-VisionAIAPI\2.CameraCalcDemo

文件夹/文件名称	说明
20220207_220418	捕获的图片。
OneCameraCal.bat	一键启动脚本。
OneCameraCal.py	单目相机标定例程。
bord-Calibration.py	内参矩阵。
ball2-Calibration.py	双小球标定例程。
Config.json	视觉传感器配置文件
Python38Run.bat	Python环境启动脚本

4.运行环境

4.1 软件要求

Windows 10及以上版本；RflySim工具链；VS Code。

> ①：若使用Pixhawk 6X飞控，平台安装时的编译命令为：px4_fmu-v6x_default，推荐PX4固件版本为：1.12.3。其他配套飞控及编译命令请见：https://rflysim.com/doc/zh/1/Hardware.html

4.2 硬件要求

笔记本/台式电脑① 1台。

> ①：推荐配置请见：https://rflysim.com/doc/zh/HowToInstall.pdf

5.实验步骤

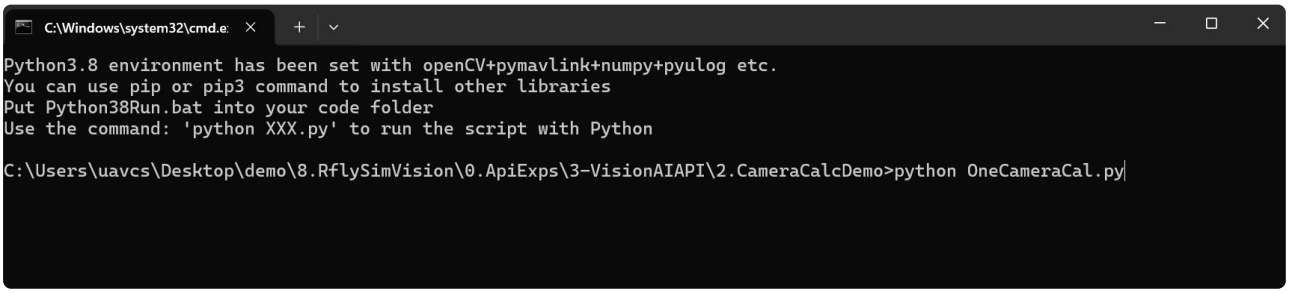
5.1 必做实验：Windows取图控制

Step 1：开启仿真

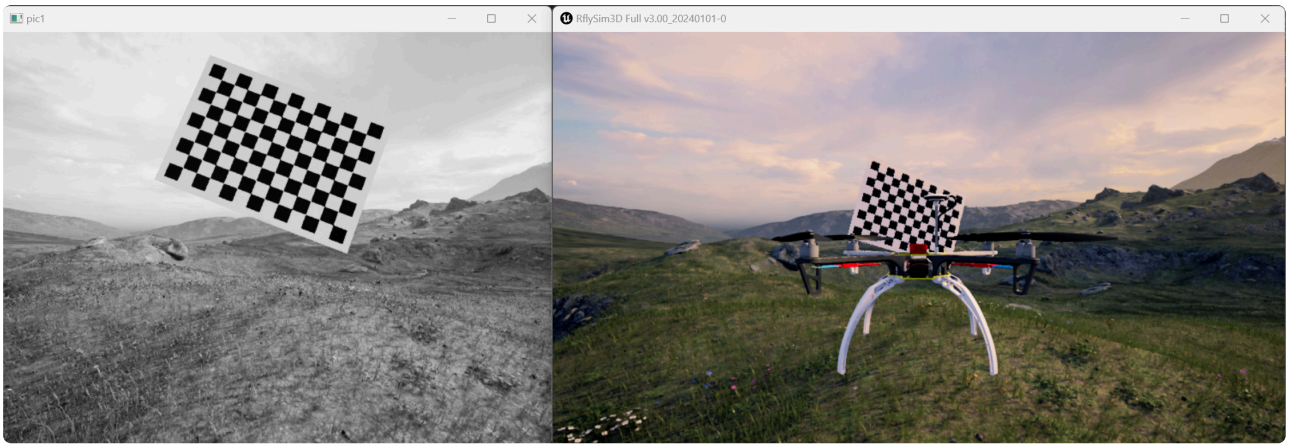
双击运行OneCameraCal.bat一键启动脚本，会自动打开RflySim3D仿真平台。

Step 2：运行控制程序

在文件夹下，双击Python38Run.bat，打开集成好的python环境，在该环境下输入`python OneCameraCal.py`运行OneCameraCal.py文件



Step3：观察结果

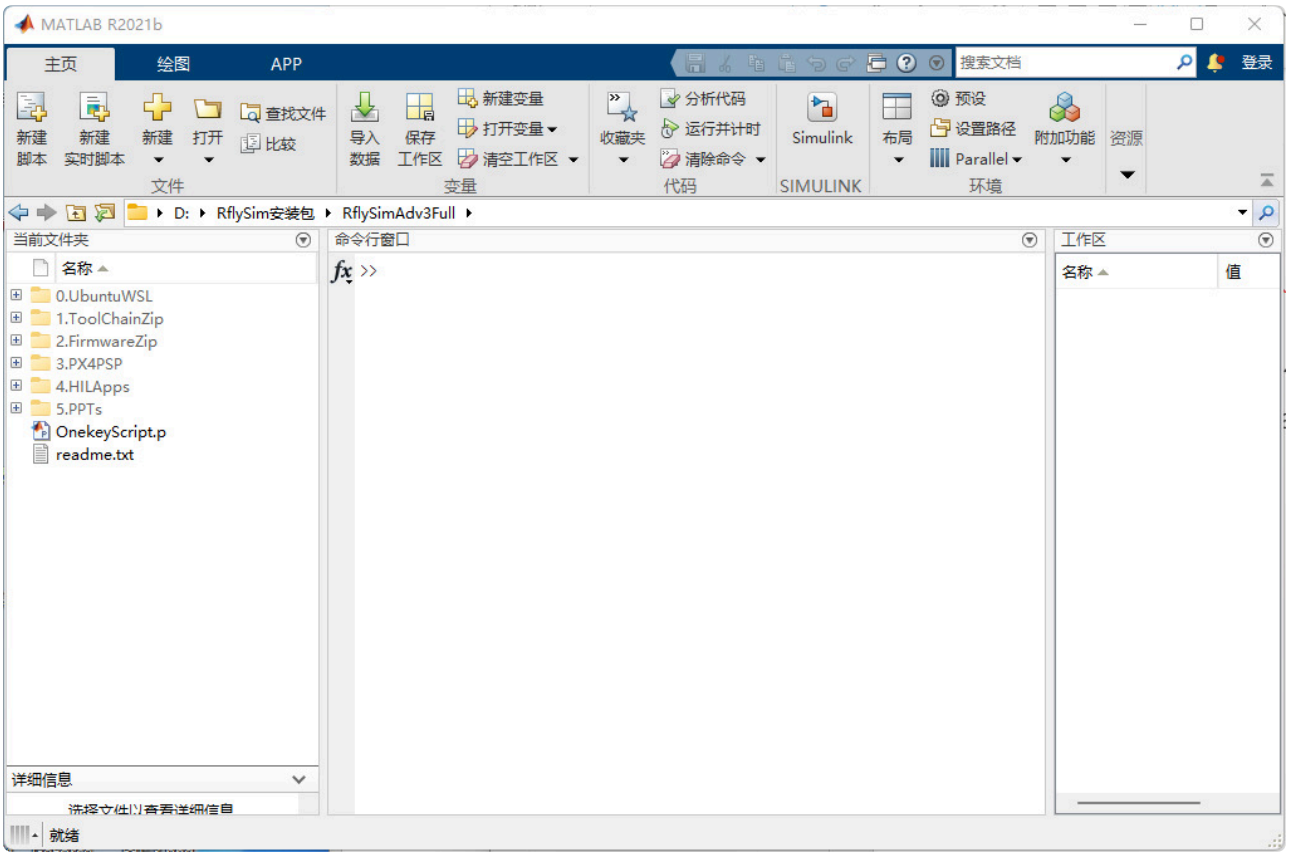


若标定板与相机的距离太远或太近，会影响检测结果，可以修改InitTargePos中的第一个数值来改变距离，获得三十张图片左右后就可将程序停止。

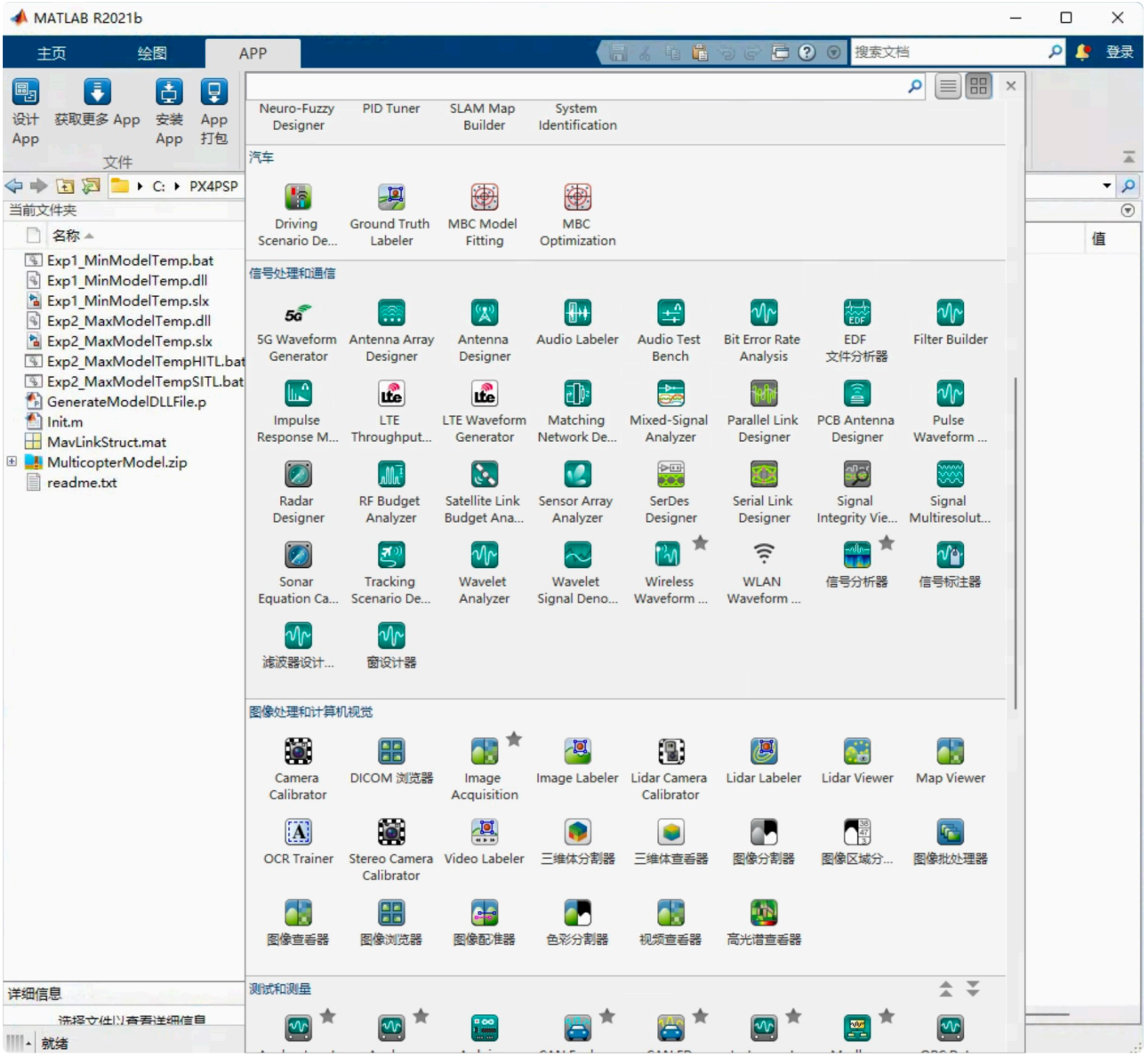
Step 4：获取相机内参

将存储下来的单目相机图像，在MATLAB中校准，得到相机内参。

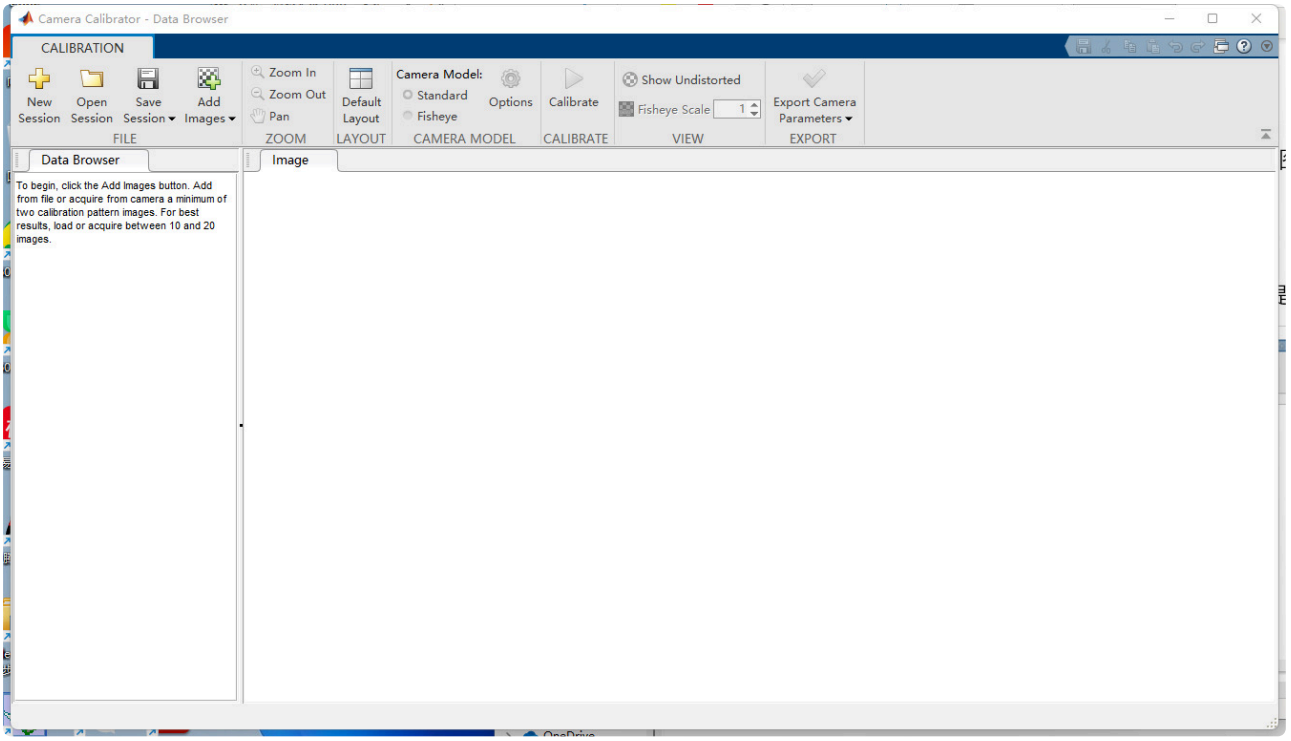
打开matlab，点击上方的APP栏。



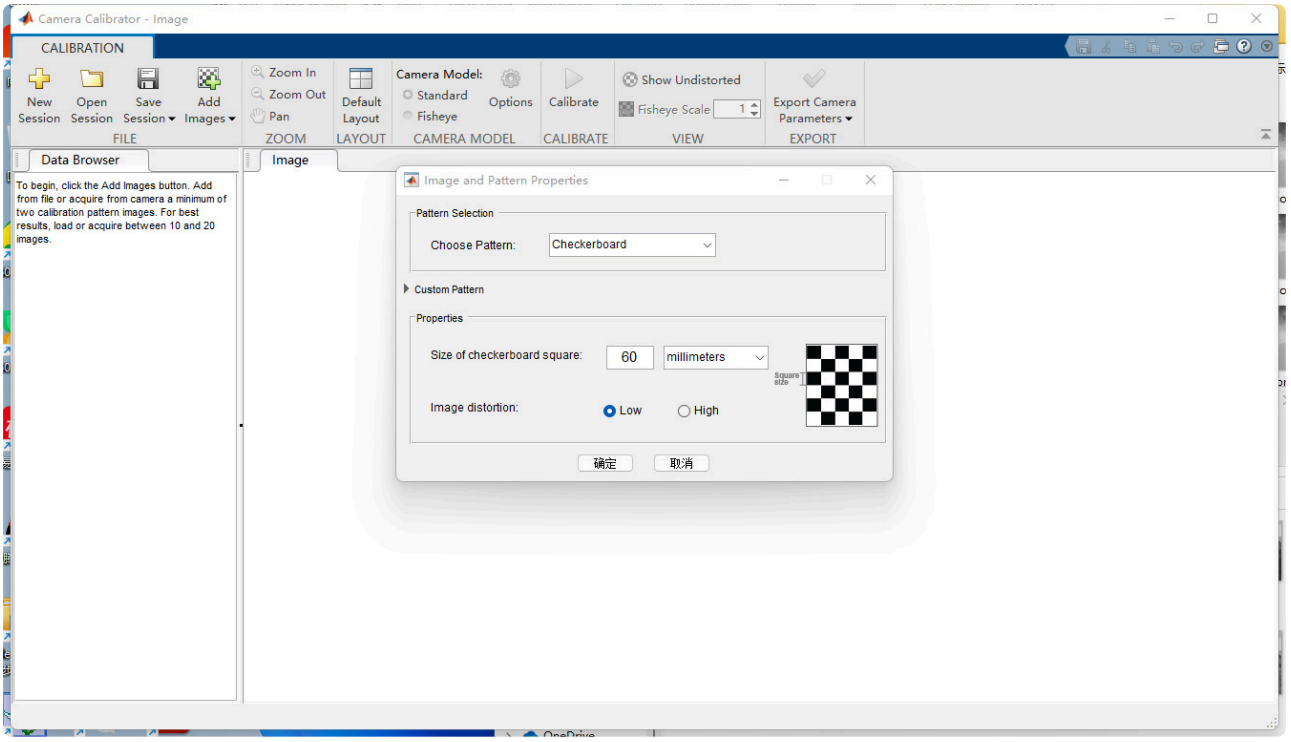
下拉工具栏，选择Camera Calibrator工具箱

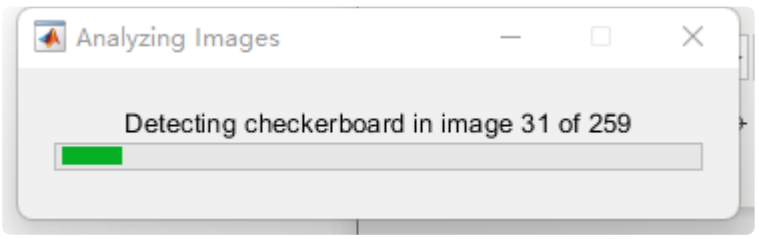


打开工具箱后，点击Add Images，打开到我们抓取图片的位置，选取图片。选取完成后点击打开按钮。



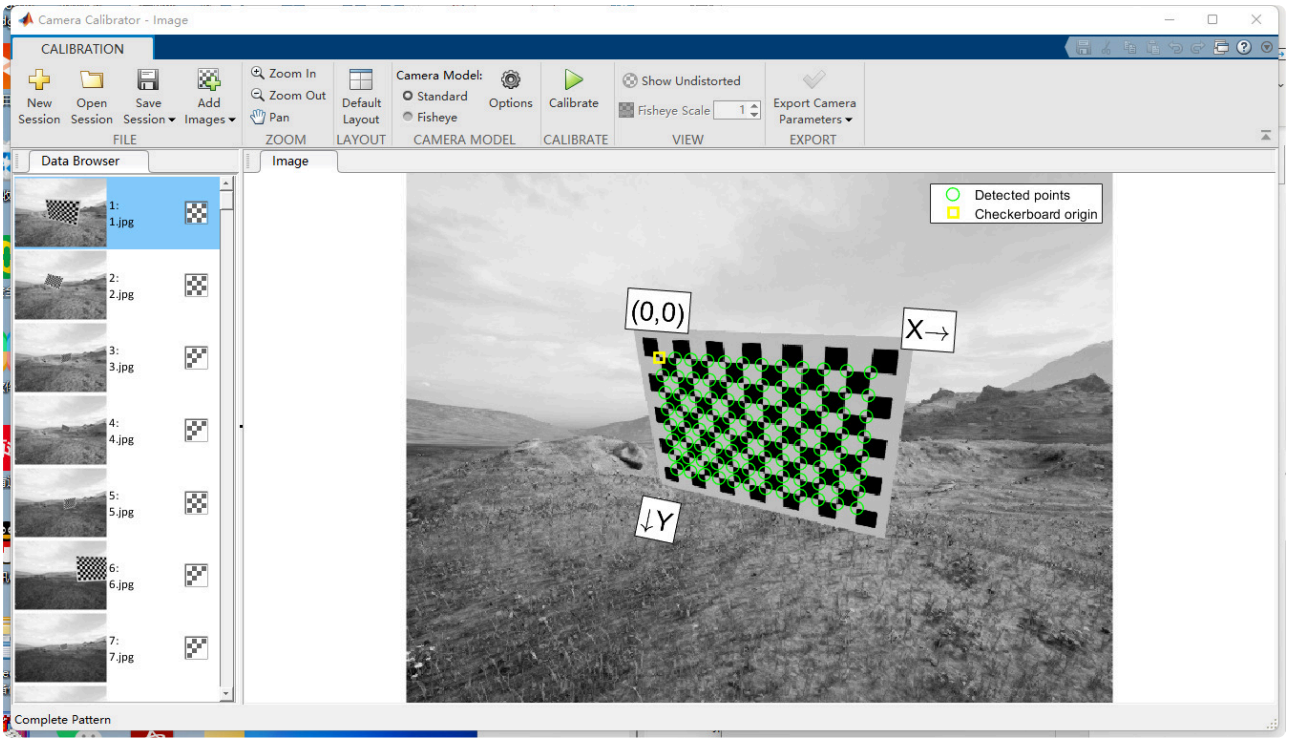
弹出下面窗口，输入棋盘格的大小。场景中为大家提供的是60mm，点击确定。



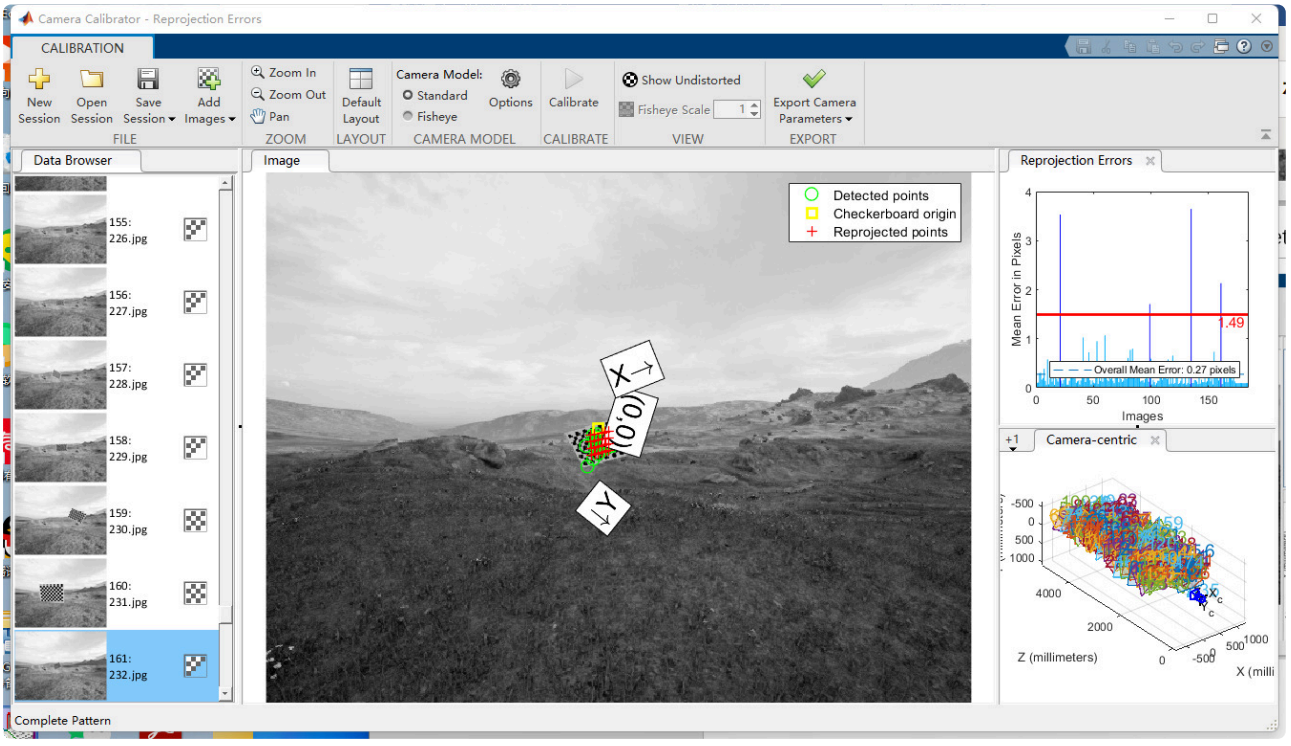


几秒钟后，工具箱会告诉你接受的图片，直接点确定。

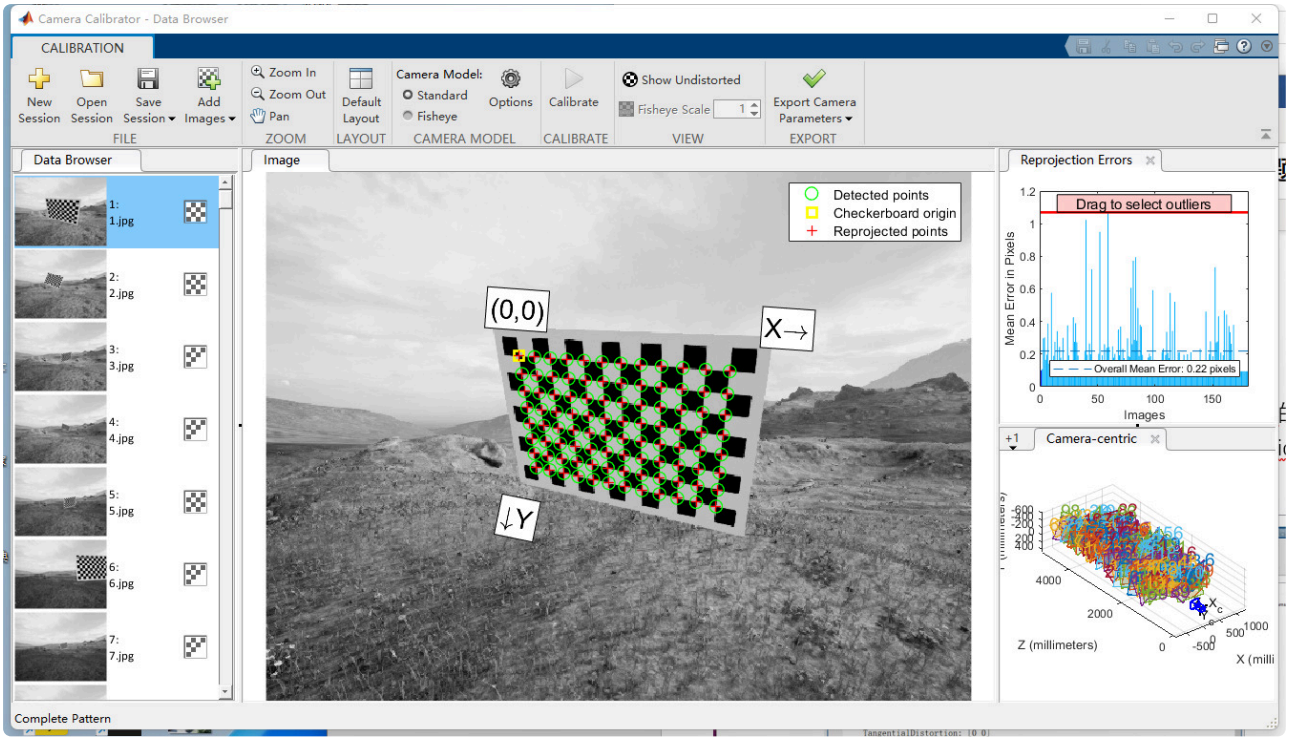
可以大概浏览一下棋盘格提取质量，应该是都在图像上。点击Calibrate按钮开始标定。



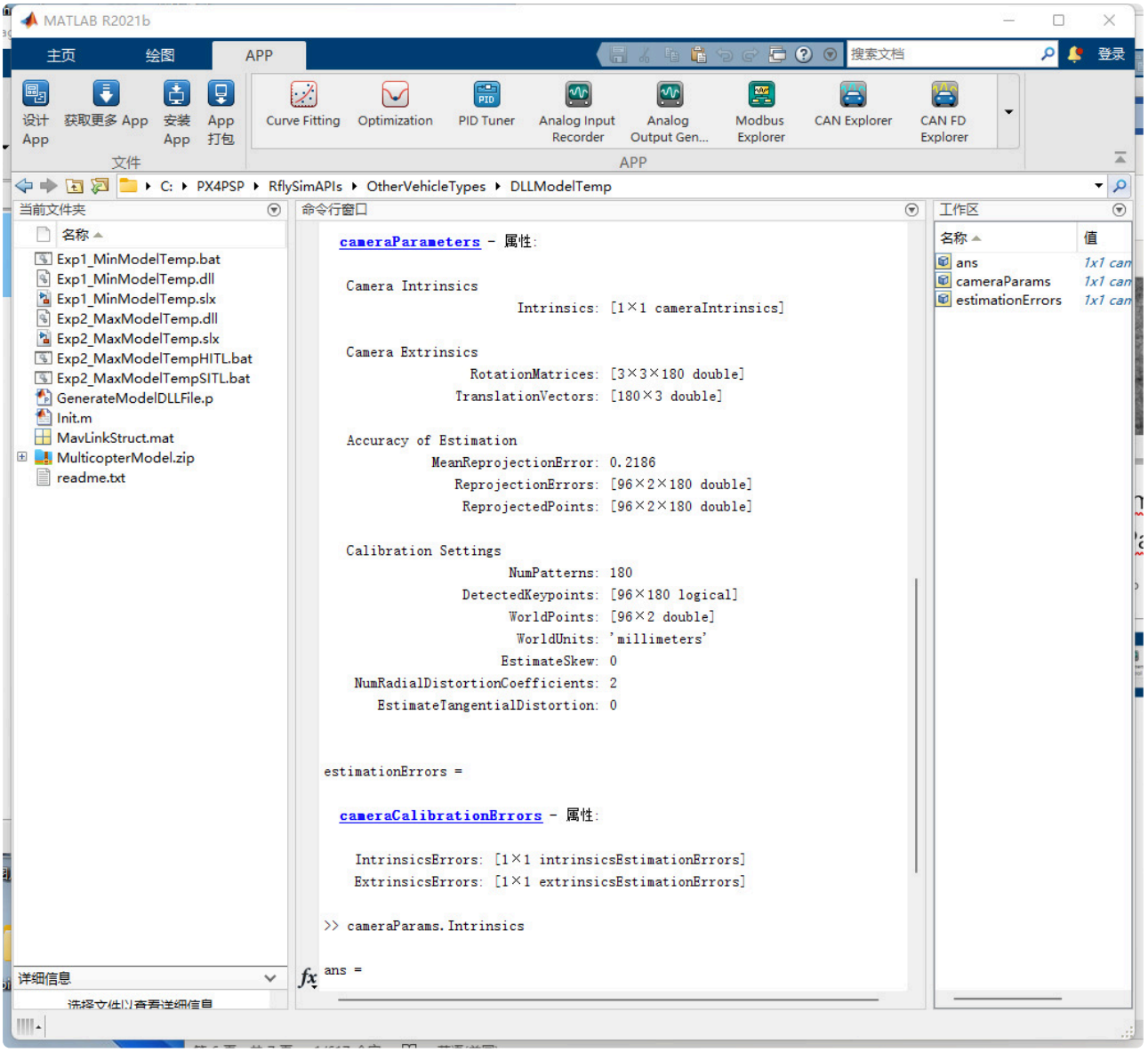
可以拖动红线选择反投影误差较大的图片，按Delete删除这些图片并重新标定。



结果满意之后点击Export导出标定的结果。



回到Matlab工作区，会看到一个名为cameraParams的结构体，双击可以看到各成员变量。也可以在下方输入cameraParams.Intrinsics查看相机信息，输入cameraParams.IntrinsicMatrix查看内参矩阵。



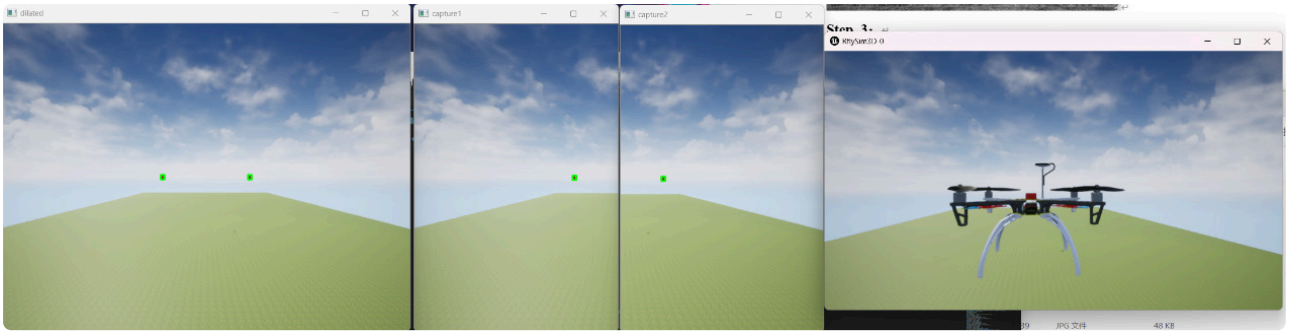
Step 5：修改文件路径

在bord-Calibration.py文件中将第19行的路径改为要标定的照片路径并运行。

```
# 图片文件所在的子目录名称
image_dir = "20220207_220418/img1"
```

Step 6：运行控制程序

在文件夹下，双击Python38Run.bat，打开集成好的python环境，在该环境下，输入`python ball2-Calibration.py`，运行ball2-Calibration.py文件。



5.2 选作实验（VS Code调试运行）

准备工作

- 先确保已经按RflySimAPIs\1.RflySimIntro\2.AdvExps\e3.PythonConfig\Readme.pdf步骤，正确配置VS Code环境。或者配置了自己的Pycharm等自定义Python环境。
- 其他步骤与上文相同，在Step2运行ball2-Calibration.py时，可使用VS Code（或Pycharm等工具）来打开ball2-Calibration.py文件，并阅读代码，修改代码，调试执行等。

扩展实验

- 请自行使用VS Code阅读ball2-Calibration.py源码，通过程序跳转，了解每条代码的执行原理；再通过调试工具，验证每条指令的执行效果。

VisionCapAPIDemo.py

> RflySimAPIs > 8.RflySimVision > 0.ApiExps > 1-UsageAPI > 0.VisionSenorAPI > 1.Came

```
ue = UE4CtrlAPI.UE4CtrlAPI()

#Create a new MAVLink communication instance, UDP sending
mav = PX4MavCtrl.PX4MavCtrler(1)

# The IP should be specified by the other computer
vis = VisionCaptureApi.VisionCaptureApi()

# Send command to UE4 Window 1 to change resolution
ue.sendUE4Cmd('r.setres 1280x720w',0) # 设置UE4窗口分辨率，
ue.sendUE4Cmd('t.MaxFPS 30',0) # 设置UE4最大刷新频率，同时也
time.sleep(2)

# VisionCaptureApi 中的配置函数
vis.jsonLoad() # 加载Config.json中的传感器配置文件
```

- 请尝试修改代码，实现飞机位置改变、相机姿态角改变、相机参数改变等功能。

6.参考资料

无

7.常见问题

Q1：无

A1：无