490Document

算法概述

相机校准

使用opencv2 包中提供的张正友标定法（A flexible new technique for camera calibration），消除相机镜头造成的畸变和误差

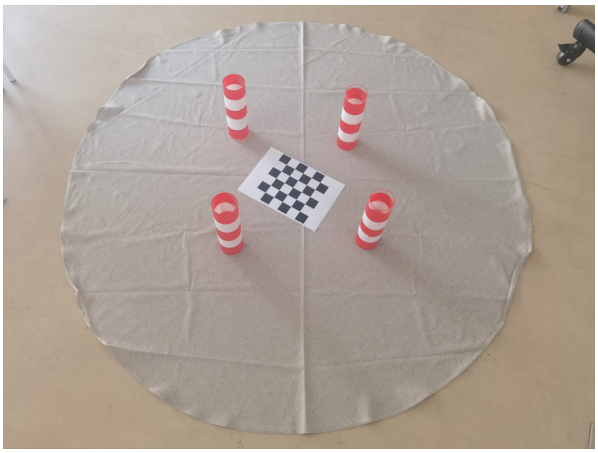
1. `cv2.findChessboardCorners()` 出角点位置
2. `cv2.cornerSubPix` 细化角点位置
3. `cv2.calibrateCamera` 通过多张图片的角点位置找出相机的内部性矩阵（intrinsic matrix）
4. `cv2.getOptimalNewCameraMatrix` 根据内部性矩阵找出对应每张照片
5. `cv2.undistort` 矫正图片

消除透视

再次利用棋盘格进行透视变换，使得图像上的点位与真实坐标对应；矫正透视后，理论上棋盘格所在平面和地角螺栓的交点，以及圆形地基在图上都已被矫正为正圆；棋盘格和方形地基被矫正为正方形；同时将棋盘格的位置对齐，便于测量坐标

用棋盘格的真实坐标与图上坐标求解单一性矩阵

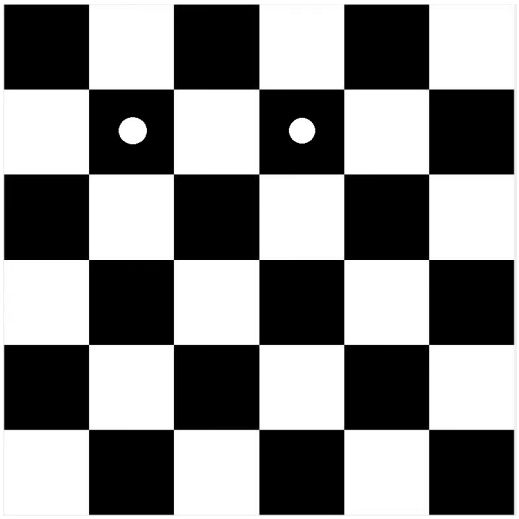
1. `cv2.getAffineTransform` 取得单一性矩阵
2. `cv2.warpAffine` 矫正透视
3. 根据 `cv2.findChessboardCorners` 中角点的相对位置旋转图像，确保矫正后的方向和原图像大致相同（为了保证地角螺栓方向不变）

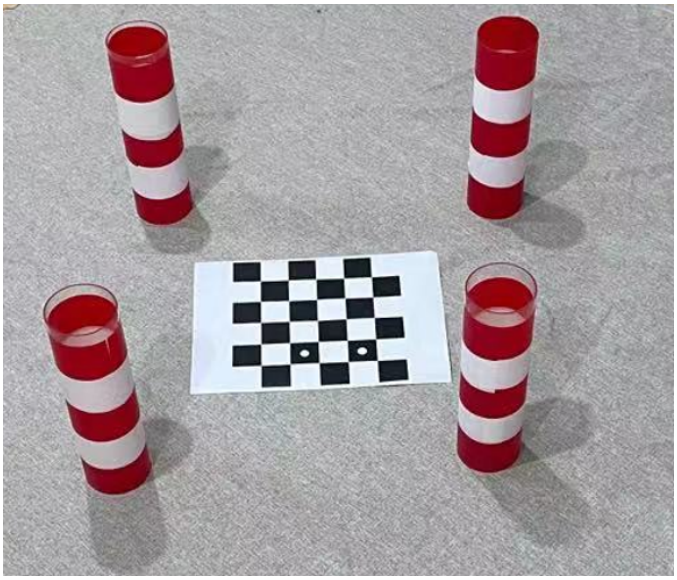


方位识别

由于6x6棋盘格是旋转对称的，因此需要用额外手段确定方位

如图：



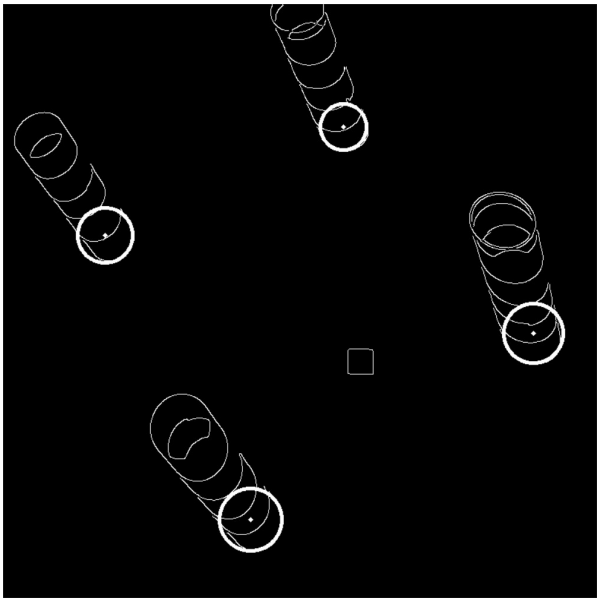


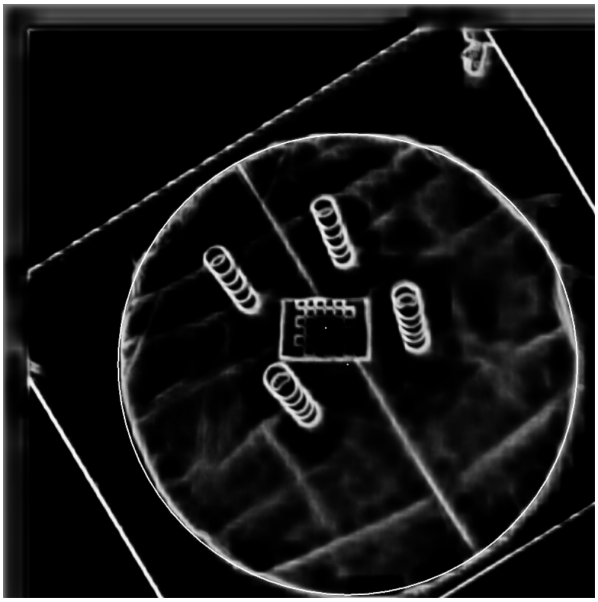
通过识别棋盘格左上角两个白点，判断方向，如图为0°，正角度为顺时针方向，未识别到时记作0°。详情可以参见Direction.py里的demo

坐标识别

1. 使用边缘识别算法识别出透视矫正后图像上的边缘
    a. 对地角螺栓：地角螺栓上贴有红白色条纹，较为明显，使用精度高但鲁棒性弱的canny算法识别
    b. 对地基：地基和地面差别较小，使用HED神经网络识别
2. 确定坐标原点：即6x6棋盘格的中心坐标，单位长度设定为棋盘格每格宽度的1/32，下方和右方为正方向
3. 对地角螺栓：使用霍夫变换识别，使用K-means算法将识别出的圆聚类，去除偏差过大的值后，因为照片中地角螺栓朝下，最低的圆即为地角螺栓的坐标
4. 对地基： 找到图像中最大的闭合边缘，其重心即为地基中心

注意：HED神经网络的输出图像边缘和原图像有偏移，进行了校准





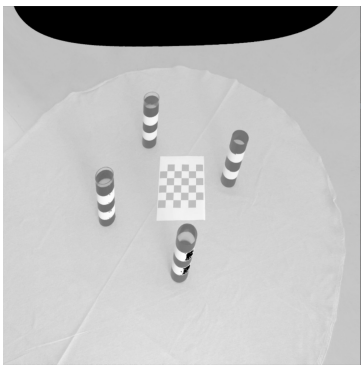
算法优化及改进

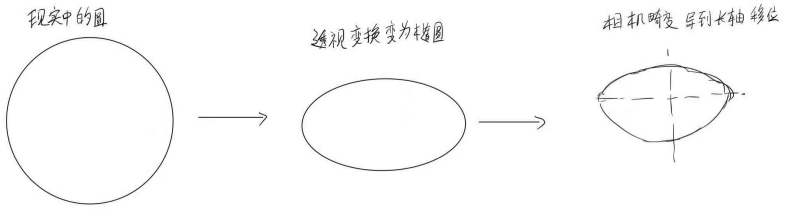
速度优化

1. 图像校准后边缘反而较为扭曲，造成识别格点的时间较长，因此可以进行适当剪裁
2. 神经网络的时间复杂度和分辨率成正比，而圆形地基为较大的目标，因此在将图像输入HED网络前先降采样，再将输出图像升采样，如图可见对圆形和中心的识别是准确

的

因为发现OpenCV的相机校准算法实际上无法校准边缘（如左图，注意左下角），导致难以按照上文所示的方法将圆形地基的像素还原为真实坐标，以下提出另一种方式：





右上图解释了相机畸变导致校准失准的原理，一方面，简单的通过使棋盘格对准并不能解决y轴畸变；但是畸变对椭圆的x轴坐标几乎不会产生影响，因此可以通过记录每张图片中圆心与棋盘格中心在X轴上的距离，和棋盘格的大致角度，根据两点最大横向距离等于平面距离，利用统计学方法求出圆心与棋盘格中心X轴，Y轴的相对坐标。

算法实现

客户端：

1. 使用socket将文件传输到服务器的指定文件夹
2. 使用watchdog库监控指定目录中新增/修改的zip压缩包
    a. 首先新建一个 `watchdog.observers.Observer` 的实例
    b. 编写事件处理函数，对应Observer中的特定的事件类
    c. 设置监听类型等参数
3. socket服务器监听指定文件夹，直到处理程序返回坐标文件，并回传客户端

网页：

使用python的Flask框架编写，触发函数返回 `app.response_class` 对象实现流式传输，在图像处理过程中实时显示处理过程

使用方法

推荐安装Anaconda（Python3.8）后在Anaconda Prompt中启动程序

配置环境

1. 解压压缩包
2. 用管理员身份打开Anaconda Prompt
3. 在conda中安装必要的包`conda install opencv flask werkzeug scikit-learn pandas watchdog numpy tqdm colorama`
4. 如有需要，可以新建环境并切换 `conda create -n [环境名]` `conda activate [环境名]`

网页版

1. 切换到脚本所在目录下，运行脚本：`python Site.py`，程序会自动启动其他脚本
2. 在命令行中输入端口，将程序返回的网址复制到浏览器（如要更改路径，更改`Site.py` 中的ROUTE变量）
3. 上传图片，等待网页上显示返回的结果

socket程序：

1. **服务器端先启动server.py:** `python server.py`**，再启动socket程序:** `python serversoc.py`
2. 第一次启动`serversoc.py`后会生成ipconfig.xml，可以将其复制到客户端的**根目录**
3. 客户端启动client后按要求输入服务器ip地址（若没有ipconfig.xml），端口，图像文件夹位置
4. 处理完毕后服务器返回一个txt文件到客户端脚本所在目录

更改参数：

1. `Server.py` 中的 `NUMBERS` 调整用于测量的图片张数
2. `Server.py` 中的 `NN_FLAG` 调整是否使用神经网络测量
3. `Server.py` 中的 `OLD_BASE_FLAG` 调整是否使用旧方法测量地基中心
4. `Server.py` 中的 `CANNY_PARAMETER` 调整canny算法的参数（仅当`NN_FLAG` 为False时生效）
5. `Server.py` 中提供了一些在服务器端查看图像处理过程的代码，去掉注释以使用

输入要求

建议上传10-15张照片，地基应位于图像中心，并且地基的轮廓全部纳入取景框内；**棋盘格不能被地角螺栓或其他物体遮挡，否则可能导致程序运行失败**

已知bug

1. 如果从命令行启动Socket脚本，客户端向服务器传送的压缩包有时会无法解压
2. 如果输入张数过少会有严重bug

函数描述

1. `local_grid(gray)` 找到四个地角螺栓中心相对于棋盘格中心的坐标
2. `HED(image,net)` 使用HED网络net寻找图像image的边缘
3. `base_grid(gray,net,corr=90)` 找到圆形地基中心相对于棋盘格中心的坐标（corr用于修正HED网络输出图像相对原图像的偏移，可以根据观察调整）
4. `base_grid2(image,net)` 根据上文中所述的改进算法找到圆形地基中心相对于棋盘格中心的坐标，**默认使用base_grid2**
5. `square_grid_NN(image,net)`，`square_grid_normal(image)`，分别使用神经网络和形态学变换找到方形地基的坐标
6. `obthomography(dst, criteria, size=800, ratio=0.04)`：消除透视
7. `direction(dst)`：使用上文描述的方法确定棋盘格方位
8. `processimage(path)`：即处理path中所有图像，将坐标记录在txt文件的主函数

改进方向

1. 进一步提高测量的精度
    a. 改进相机校准的算法（使用OpenCV文档上提供的案例有较大误差）
    b. 改进边缘识别算法（使用BDCN等更高性能的神经网络/Canny算法自动调整参数）
    c. 改进物体中心识别精度
2. 用户界面

a. 为网页端写一个专用的处理函数，去掉监听环节改善性能

b. 让网页端能看到处理过程信息和报错

c. 增加网页端调整程序参数的功能

d. 修复socket程序的bug

附：测试误差

Name	Actual	测试结果（旧方法）	测试结果（新方法）
450x420	32-35	102.96	32.57
500x500	20-22	27.85	18.28
550x550	10-12	75.7	45.21

更新内容

1. 基本解决棋盘格角点识别速度慢的问题：由于土壤和鞋子等噪声的影响，角点识别速度很慢，因此在每次识别角点前，先进行自动截图处理，将噪声因素去除，并在识别角点后还原原图与坐标。在此针对性地提出几点拍摄要求：
a. 棋盘格需要打印得更清晰，要求黑白对比度高。
b. 上述截图操作默认截取横向1/4到3/4，纵向1/3到2/3的部分，因此，需要大概目测棋盘格在该范围内。
c. 主柱需被完全拍入图片中，图片边缘不可与圆形主柱相切。
2. 网页优化（尚未完成）：使网页的显示内容更人性化。