# 环境配置

目前的测距方法使用YOLOv5及OpenCV实现，如果没有经验，详细环境配置方法可参考：[myosotis0v0/yolov5-train-process (github.com)](https://github.com/myosotis0v0/yolov5-train-process)

环境配置完成后，能正常运行Detect\_Tag\_v2.py，并显示检测结果即复现成功（注意修改代码中的几个绝对路径为自己电脑上项目的对应路径，或改为相对路径）。也可以将目标图片文件夹路径改为/dataset下的其它文件夹，观察不同结果（使用了两种相机拍摄，图片质量有所不同）。

# 原理

## 架构

定位测距系统由计算机、相机、镜头、标签等组成，可以实现实时获取图片并获取位姿信息。整体架构如图1所示。

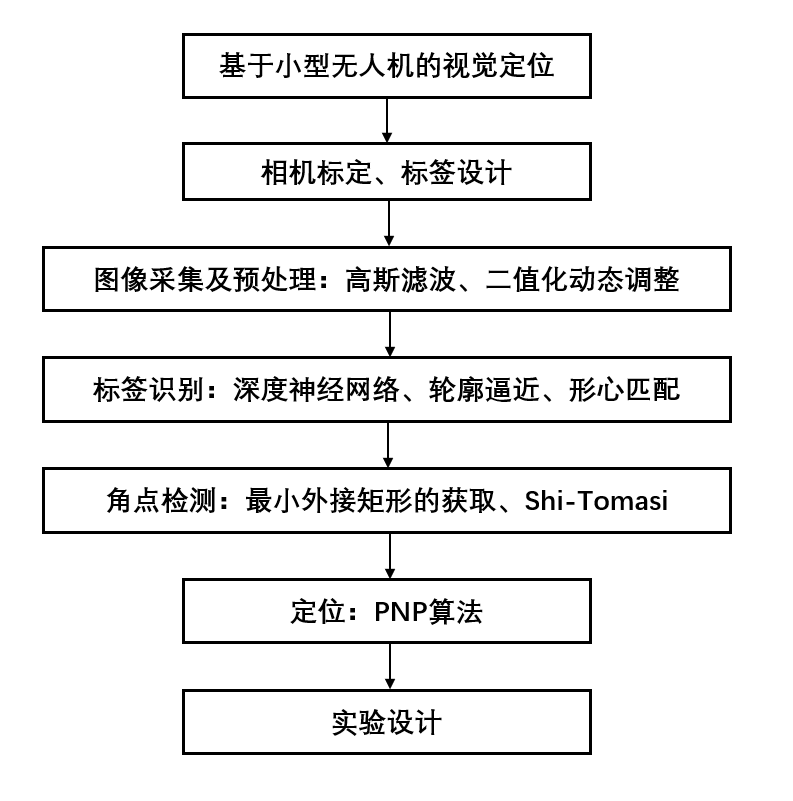


图 1 定位系统架构

## 标签识别

为对无人机进行精确地位姿测距，一个便于识别的特征标签必不可少。首先，标签要包含足够的信息，使计算机视觉系统可以利用信息识别标签是否在视野中，并排除周围环境的干扰，同时有利于计算机视觉系统提取标签的三维相对位置信息。其次，特征标签应该比较简单，使计算机视觉系统能够相对快的速度处理图像。最后，标签的设计要充分考虑在近距离视野范围受限的情况以及相对较远的距离的情况也能发挥作用。图2是目前使用的标签。

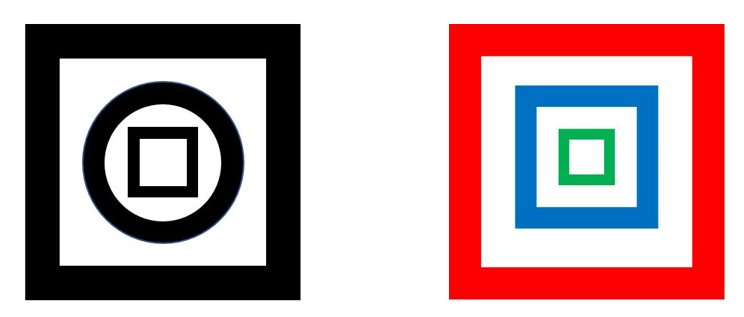


图 2 设计的标签

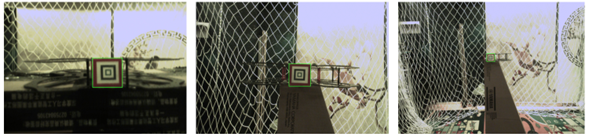
对彩色标签，首先采用YOLOv5深度神经网络粗略提取标签在图像中的位置信息。通过训练好的网络对所获得的图像进行识别，获得标签的预检测范围，可以很好地排除干扰点，缩小检测范围。图3中绿色框为水平距离20cm、50cm和150cm下YOLOv5目标识别网络对彩色标签的识别结果。

图 3 YOLOv5标签检测结果

YOLOv5目标检测的矩形框为后续轮廓逼近算法提供了预检测范围，使得在这个范围内我们能够通过轮廓识别来查找方形轮廓。在预处理后的图像数据上，借助OpenCV的findContours函数，它返回每个轮廓的点向量以及拓扑信息。

在函数中限定了RETR\_CCOMP条件，这将使所有轮廓仅建立两个层级，即内层和外层轮廓。通过这种方式，我们可以更好地识别轮廓的结构。对于提取到的不同轮廓，我们可以通过比较轮廓的面积信息，精确识别并确定轮廓的位置。在这里，我们特别关注红色内侧矩形的提取，确保提取的轮廓特征位置一致。

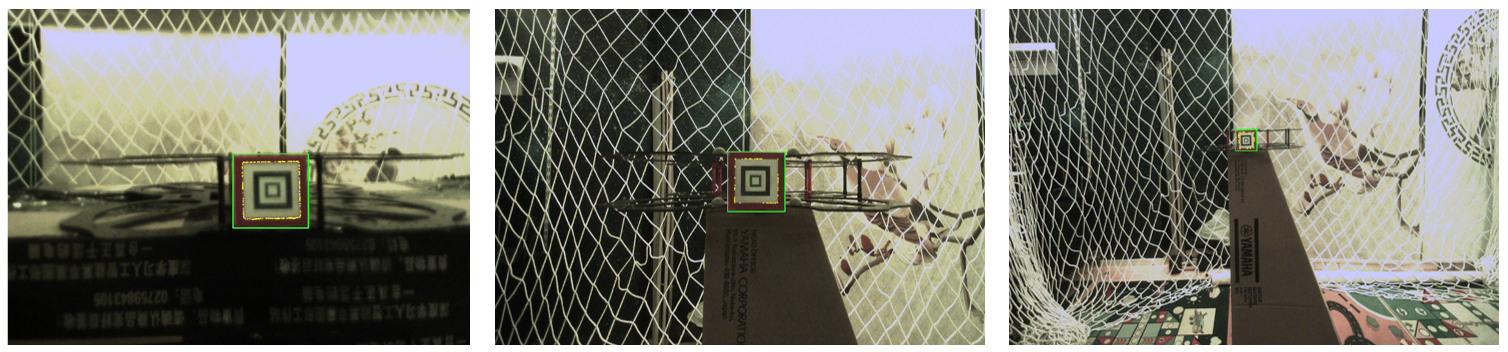
最后，在识别到的轮廓上，我们通过minAreaRect函数获取轮廓的外接矩形，得到矩形标签的四个角点。该外接矩形作为识别与测距的关键特征信息，有助于准确地定位标签的位置。轮廓提取结果如图4中黄色线所示。

图 4 轮廓提取结果

由于传统算法鲁棒性限制，并非所有场景都能成功提取轮廓（实际测试时成功率并不高），因此在提取失败时直接采用YOLOv5目标检测框作为标签轮廓，用于后续计算。

## 位姿解算

完成目标识别后，最后一步则是利用标签的尺寸信息、图像中提取的特征点的位置和相机内参数来解算相机坐标系和像素坐标系之间的相对位置和姿态关系，即位姿测距。这里采用PNP算法，原理图如图5所示。

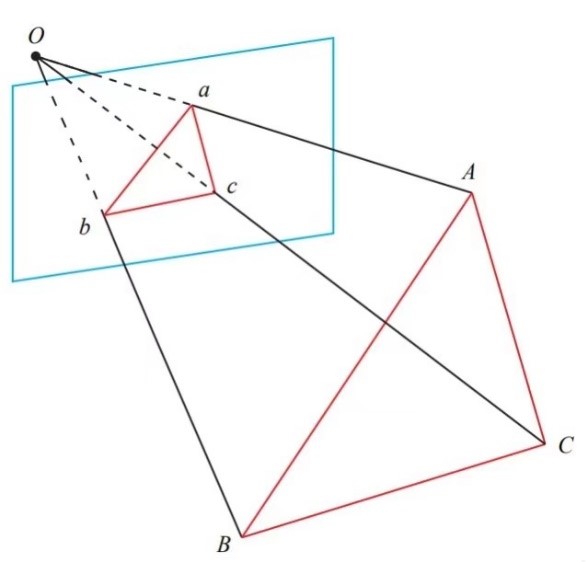


图 5 PNP算法原理图

具体实现中，调用OpenCV中的solvePnP方法进行计算。输入参数为标签的四个角点以及相机的内参，见Detect\_Tag\_v2.py的line178。

# 精度

在如图6所示的光学平台进行多次实验，得到的结果如表1所示。

图 6 测距实验使用的光学平台

表1 彩色标签位姿测量误差

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tag与相机距离（z轴） | 35cm | 55cm | 75cm | 95cm |
| x轴误差 | -0.36cm | -0.40cm | -0.57cm | -0.72cm |
| y轴误差 | 0.25cm | 0.33cm | 0.39cm | 0.34cm |
| z轴误差 | -0.96cm | -0.74cm | 0.37cm | 2.03cm |

在1m内的X、Y轴平均误差都能达到毫米级，Z轴误差在水平距离接近1m时可能超过1cm。