无名科创 TIVA 开源飞控
使用说明书 V1.0.0

无名小哥 编著

目录

一、无名科创简介

无名科创团队是一支主要由一群工科技术男组成，秉承“厚德博学、崇实去浮”的校训，开启了一路打怪升级进阶逆袭之旅，无名科创团队是一个敢于探索、积极分享的团队。我们欢迎所有广大电子爱好者、喜爱编程与创新、飞控 DIY 的朋友们的加入。微信公众号：namelesscotrun，无名科创飞控技术讨论群：540707961。

公司官网：www.nameless.tech

客户使用心得反馈、意见征集贴

http://www.openedv.com/forum.php?mod=viewthread&tid=286467&extra=page=1

知乎改进意见征集帖：https://zhuanlan.zhihu.com/p/54471146

技术博客：http://blog.csdn.net/u011992534

优酷个人频道：http://i.youku.com/u/UMTc3OTMyMjM3Ng==?spm=a2h0k.8191405.0.0

教学视频 B 站主页：https://space.bilibili.com/67803559/#/

欢迎关注无名科创微信公众号与

知识星球会员圈

提供免费技术问答服务!!!

方式 1、微信扫一扫

上图中：左为公众号、右为知识星球会员圈

方式 2、微信公众号搜索：无名科创

或者 namelesscotrun

为什么选择无名创新：

感动人心价格厚道　最靠谱的开源飞控

国内业界良心之作　最精致的售后服务

追求极致用户体验　高效进阶学习之路

萌新不再孤单求索　合理把握开源尺度

响应国家扶贫号召　促进教育体制公平

新时代奋斗最出彩　建人类命运共同体

二、团队的发展于技术创业之路

无名团队发展历史：多旋翼飞行器飞行控制系统（简称飞控）是我们团队历届主研项目，团队 13 年即开始第一代飞控的研究，从最开始的小四轴，到后来的多旋翼飞控，经历 N 个版本改进，经历无数次断桨、射桨、炸机，一步一步完善与改进，整合除了目前我们这款对外开源的飞控。团队历来贡献者均就职于无人机公司做算法相关工作。目前我们的飞控更加完善，更加稳定，更加适合学习，主要核心代码自写率达到百分之 90 以上，代码基本上是逐行注释，整个飞控框架清晰明了，模块化封装规范，方便大家学习与二次开发。由于作者目前仍然在校，主研项目仍为飞控，个人时间比较多，可和大家一同交流学习。

开源飞控技术创业之路：考虑到对飞控学习而言，国外商业级飞控 APM、PX4/Pixhawk、Autoquad 等平台不适合初学者入门学习。而国内传统飞控如圆点博士、烈火、Crazepony、ANO、Light、INF、恒拓等随着创始者们相继毕业、商业转型、工作时间紧，代码缺乏更新维护（涉密），以及技术支持不及时等，随着飞控整个行业的技术成熟， 对从业者的要求越来越高，在国内开源飞控“青黄不接”的当下，2017 年 9 月，创立了无名科创开源飞控团队，决定将个人自研的第一代飞控。NamelessCotrunQuad_V1.0 进行了程序模块化、规范封装等优化，主控板与 IMU 采用分离式设计，对 IMU 模块进行独立减震处理，主控选用 STM32F103RCT6，气压计选择低成本高性能的 SPL06（精度媲美 MS5611，价格为其 1/5）、磁力计选用业界新款 IST8310（DJI Mavic 同款），极大的降低了飞控学习成本，结合配套持续更新的教学视频与技术博客进行飞控相关技术的讲解，赢得的广大飞控爱好者的青睐与支持，前期自己手工贴片的 50 套飞控学习板，不到一个星期就已卖断货， 管理的无名科创开源飞控技术交流群（飞控行业第一个交流 5000 人企业认证群）成为当前学习层面，公共交流、活跃度超高的群。

我们的服务宗旨是：打造国内功能最多、性能最好、成本最低、可玩性最强的开源飞控学习平台。帮助大家以最小的代价、最大的获得感、最快的速度、最便捷的进阶路

线学习飞控相关算法，顺利完成进阶逆袭！！！

我们的愿景是：一批批的学习爱好者能共同贡献出自己的力量，帮助新人顺利完成进阶逆袭之路。前人栽树，后人乘凉，生命不息，奋斗不止。立志为行业壮大有生力量，培养出更多把无人机当作毕生奋斗事业去做的人，切勿只因比赛证书、短期毕设、造数据、水论文而浅尝辄止！！！

三、无名创新创作 TI 飞控的初衷

随着无人机行业的发展，以及各大开源飞控平台的推动，使得行业草创初期，行业大佬前辈们需要踏实苦干、执着坚守、花大气力研发出来的飞控算法与功能，今天的我们可以基本不费力的通过自己学习相关算法编程去实现。十年前，国内无人机从市场规模和从业人数上来看，压根算不上一个行业。国内早期的飞控研发者，大多是来源于电子、控制、计算机、航天相关专业优秀人才，他们对飞行有着与生俱来的爱好与执着，对技术追求极致，他们作为行业的拓疆者，从最早起的航模论坛给发烧友出售飞控的盈利探索时代（如极飞、大疆），到现在消费级无人机和工业级无人机遍地开花的局面。独立拥有成熟飞控技术的企业越来越多，当前工业级无人机已摆脱了消费级无人机的影子，开始出现产业逐渐明晰、需求不断涌现，技术标准逐渐成型、市场不断细分的形势。无人机厂商已经不仅仅关注于无人机本身，而是提出了针对多个行业的整体化解决方案。无人机行业处于成长期，未来发展前景广阔。

全国大学生电子设计竞赛是教育部和工业和信息化部共同发起的大学生学科竞赛之一，是面向大学生的群众性科技活动，属于 A 类赛事，含金量很高，高校普遍对本竞赛比较重视，同时赛题可选性很大，使得各个层次高校学生都有机会参与进来，竞赛控制类题目从 2013 年起，开始四旋翼飞行器相关试题。实现的任务从早起的一键自动起飞、位置悬停、寻迹、投放物块、追踪移动小车、穿越障碍、模拟灭火等。视觉部分传感器处理从最开始的线性 CCD TSL1401 到现在的必须光流、数字摄像头/OPENMV 全上等。这无疑是对参赛选手的能力极大挑战，学生需要很长时间准备，前期花大量精力学习和积累飞控相关知识以及视觉知识，能高效熟练解决无人机在参赛过程中的各类问题。

经过多方调研，当前大学生电赛飞行器中存在如下问题：

A. 未来十年国赛赞助单位均有 TI，参赛的飞行器主控必须使用 TI 的单片机，

需要处理传感器采集、数据滤波、姿态解算、惯性导航、控制算法等所有内容，之前 STM32+TI、STM32+APM/PX4 模拟遥控器信号的伪操作方式行不通了。无疑加重了学生的参赛工作量，使得能力一般的学生稳稳的拿省二、省三的机会都没有了。指导老师不懂飞行器相关原理、技术要点，无实际指导效能；

B. 制作飞行器开销巨大，经费不充足的单位，需要学生自己先行垫付，开始很肉疼，最后直接烧钱到麻木，要么弃赛，要么变无人机真爱粉，最终修炼大神

C. 学生自学，学长帮带的形式在有些时候行不通，能力一般的师兄毕业不从事无人机相关工作，技术指导有限，有能力的师兄毕业去了 DJI，签了保密协议，无法指导

D. 任何游戏都怕人民币玩家单位，经费充足的 RMB 玩家直接私下下找相关公司定制带 Logo 的整机参赛，相关公司公开宣传对应方案，售价动不动就 2~3 万，这些公司所为有违竞赛公平，极大的激发了民怨，拥有飞行爱好梦想的学生，看到这样的飞机出现在比赛现场击败自己时，只能默默抹眼泪。

E. 市面上目前没有一款基于 TI 主控的低成本开源飞控给大家学习，相关技术团队与公司不愿意出低成本学习方案，因为微薄的利润支撑不了情怀与工作量，养活不了一个技术公司，所以作为现实需求，只能是像早期消费机那样挣取暴利快速致富实现转型。

F. 国内开源飞控鱼龙混杂、问题颇多，但是相比三年前一抹黑学习飞控的时候，现在所处的情形，要好太多太多，大家对国内开源飞控多一些包容和理解，多提宝贵意见，拍砖打脸没有关系，虚心接受各方批评指正，促进圈子、产品健全与完善，我们会努力承担更多的责任和担当。

G. 设计 CarryPilot 与 GankerPilot 初衷是充分考虑当下国内高校飞控类竞赛学习现状，经过了多方调研考察，我们合理把握开源尺度，秉承道义，经过半年的研发，推出了目前两款基于 TI 平台的飞控，飞控算法部分会提供讲解视频，功能模块具体实现讲解，独家飞控调参视频，视觉方向教程，提供全程售后技术支持，无名创新旨在打造一款公平的、靠谱的、高效学习的竞赛飞控学习平台。

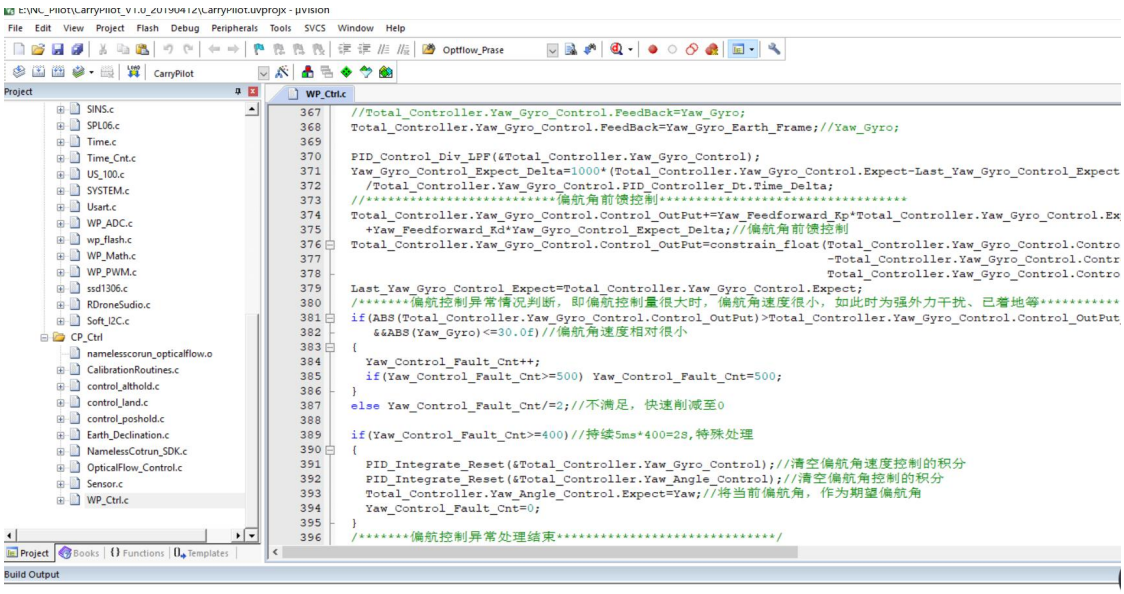
H.飞控支持对家地面站，山外多功能调试助手，匿名地面站，其中飞控的遥控器校准、电调形成校准、磁力计椭球校准、加速度 6 面校准、水平校准等可直接通过操作遥控器实现，极大的方便客户使用，飞控代码采用 Keil 开发，方便国内编程初学者的使用。

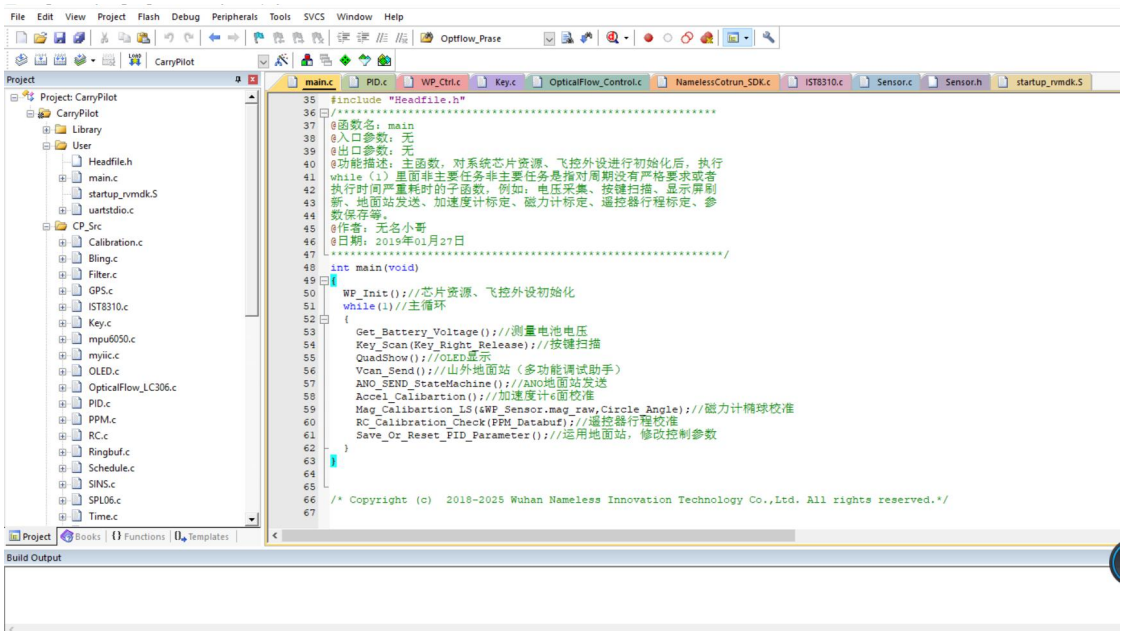
I.主控平台 TM4C123GH6PM，飞控预设 8 路 PWM 口、6 路通讯串口、两路 I2C 口、4 个预留 IO 口、4 个可编程状态指示灯、2 个板载按键、1 个 OLED 显示屏 SPI/I2C、1 个外部电压检测口、1 个 SWD 下载调试口、飞控功能和慧飞者基本一致：

- 一键起飞
- 一键着陆/GPS 返航降落
- GPS 定点/定速巡航
- 光流定点悬停/刹车
- SDK 开发者模式（直接通过串口数据帧控制，也可以事先设定轨迹）
- OPENMV 循迹、追踪移动物体。
- 更多.....

三、飞控涵盖的主要算法介绍

无名科创自研飞控平台，经过武汉科技大学连续四届研究生师兄们参考国内外主流飞控（APM、Pixhawk、Autoquad、MWC、CC3D、INF、ANO、LIGHT 等）的算法与整体框架的进行深入学习基础上，经过软、硬件的精心设计，继承与发展，目前飞控整体功能相对完善，主要功能有：姿态自稳、超声波、气压计定高，低空光流定点悬停、户外 GPS 定点，GPS 模式下定速巡航、一键返航着陆等功能，涵盖飞控学习主要核心算法：传感器滤波（一阶 RC、二阶巴特沃斯、带阻滤波、陷波滤波等）、姿态解算（梯度下降、DCM 等）、惯性导航（三阶互补滤波、单观测量卡尔曼滤波、双观测量带延时修正的卡尔曼滤波）、传感器校准（6 面校准、最小二乘法球面拟合、中心校准等）、控制器结构与算法设计（PID 控制器、微分先行控制器、前馈控制器、自抗扰控制 ADRC）等，本飞控在不同机型、动力装下均有测试，测试机型有 QAV250、F330、F380、F450、S500、T650 等，能够满足不同需求的人群使用。





3.1 滤波器设计

针对传感器内部低通后输出数据具有滞后性，本项目中，对传感器采集时不设置内部低通，比较原始信号在四旋翼悬停油门附近与静止时的频谱图，确定因震动产生的传感器噪声截止频率，设计二阶巴特沃斯滤波器对噪声信号进行处理。同时针对姿态解算与惯性导航主导传感器不同，本项目中同一量在不同使用条件下（姿态解算、惯导、反馈、校准等）的截止频率也不一样。

3.2 姿态解算

利用互补滤波、梯度下降完成四元数的更新，经欧拉角转换得出姿态的俯仰和横滚角。本项目中，避免磁力计干扰引起水平姿态角解算错误，磁力计不参与水平姿态矫正，直接采用一阶互补滤波算法获取飞行器的偏航角。

3.3 惯性导航

根据加速度计比力模型，将三轴加速度计原始数字量，通过在载体系统到导航系下的旋转矩阵归一化处理后，得到导航坐标系下比力加速度，减去重量加速度后，得到用以导航的运动加速度，本项目前期对 APM 飞控采用的基于经典回路反馈法的三阶互补方案原理进行了论证与设计实现，接着对竖直位置、速度采用单观测量的卡尔曼滤波进行惯导融合，接着针对气压计观测传感器滞后性，导致快速运动时，惯导收敛慢问题，提出了一种带延时修正的惯性融合算法。目前水平方向惯导融合采用的是双观测量带延时修正的卡尔曼滤波惯导算法，已实现户外 GPS 定位下的目标点悬停（相关算法已开源，见个人 CSDN 博客）。

3.4 控制器与算法设计

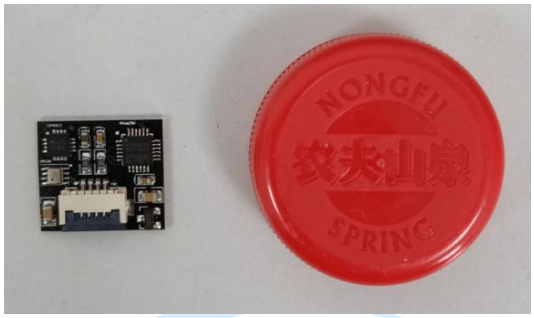
参考 APM、Autoquad 整体控制结构基础上，飞控具有自稳、手动、定高、低空光流辅助悬停刹车、GPS 定点、GPS 模式下定速巡航、一键返航等模式，其中姿态环采用的是双闭环控制，角度环采用单比例 P 控制，内环加速度采用 PID 控制，其中内环微分项采用的基于 20Hz 低通后的微分信号，避免了噪声对控制器微分的不利因素，增强了控制系统的鲁棒性。水平位置控制采用：水平位置+水平速度+水平姿态（角度、角速度）四环串级控制，竖直高度采用的是：高度位置+竖直速度+竖直加速度三环控制，针对传统飞控大多数只有两个控制环，本项目中加速度反馈信号采用的是 5Hz 截止频率数字低通后得到，避免了加速度噪声引起控制系统的异常输出引起油门突变导致的失控。另提供设计好的姿态角速度控制器的自抗扰控制代码供大家学习。

移植了开源飞控 APM 里面加速度六面校准算法，并对算法里面的牛顿消元法进行了推导。磁力计校准移植开源云台项目里面基于最小二乘法的椭球在线拟合，校准后的磁偏角可以做到+-60 度倾角内，磁偏角误差不超过 5 度。飞控将加速度计校准、磁力计校准模式直接结合遥控器动作位予以实现，方便操作。

四、飞控硬件介绍

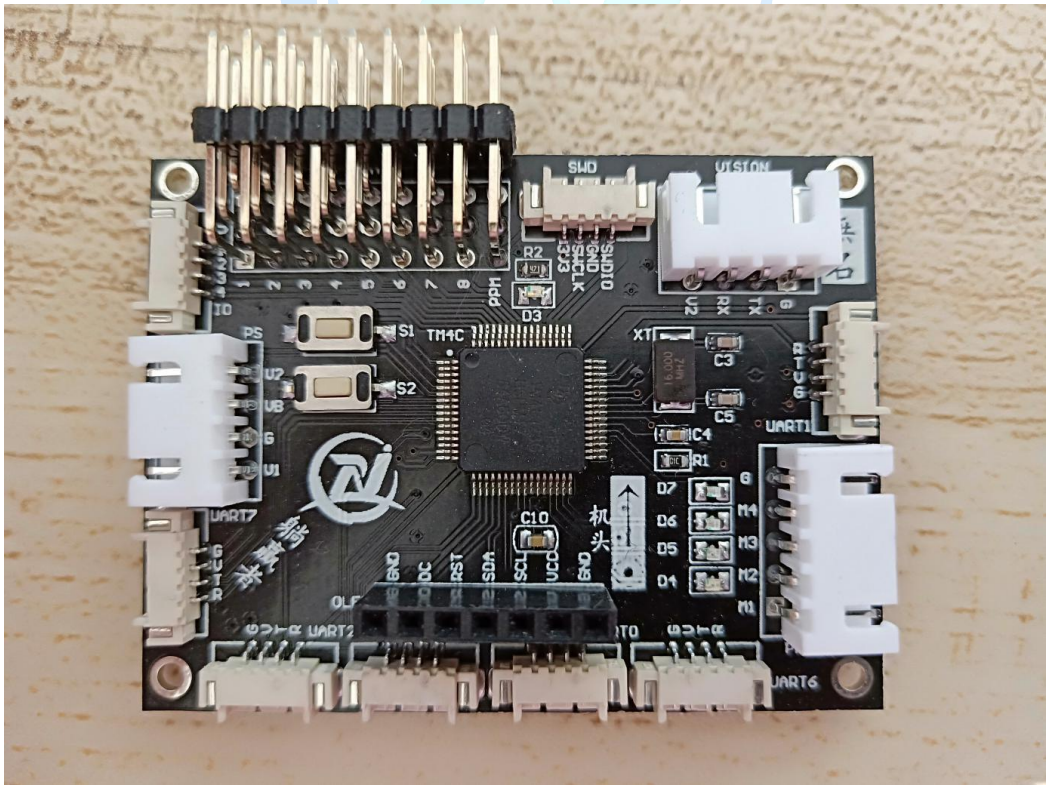
4.1 传感器

独家首创 10 轴 IMU 组合：MPU6050（加速度计、陀螺仪）+IST8310（DJI 同款磁力计）+SPL06-001（业界新宠歌尔高精度气压计、媲美 MS5611）。



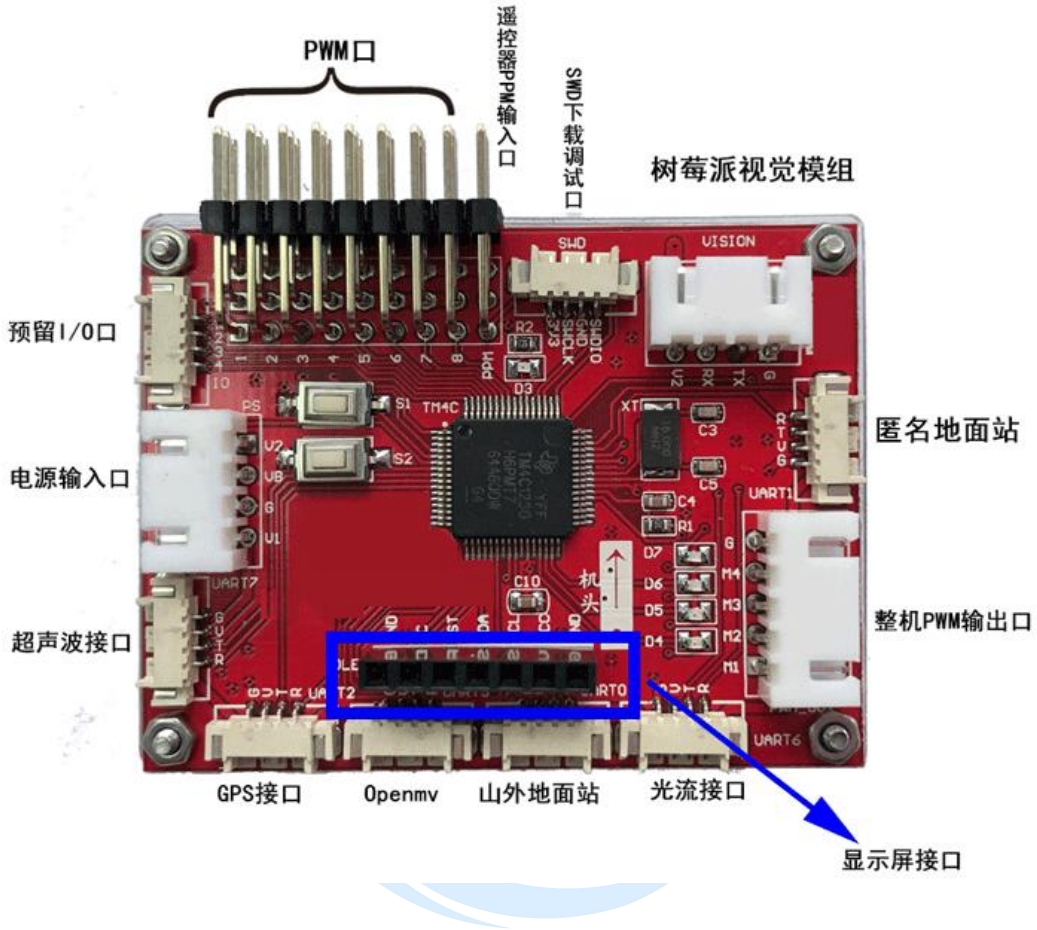
4.2 飞控板

飞控 MCU 为 TM4C123GH6PM，这是一款强大的飞控，主频 80Mhz，程序存储器（Flash）容量是 256KB，RAM 容量是 32K，自带 2K 的 EEPROM，支持单精度浮点运算。



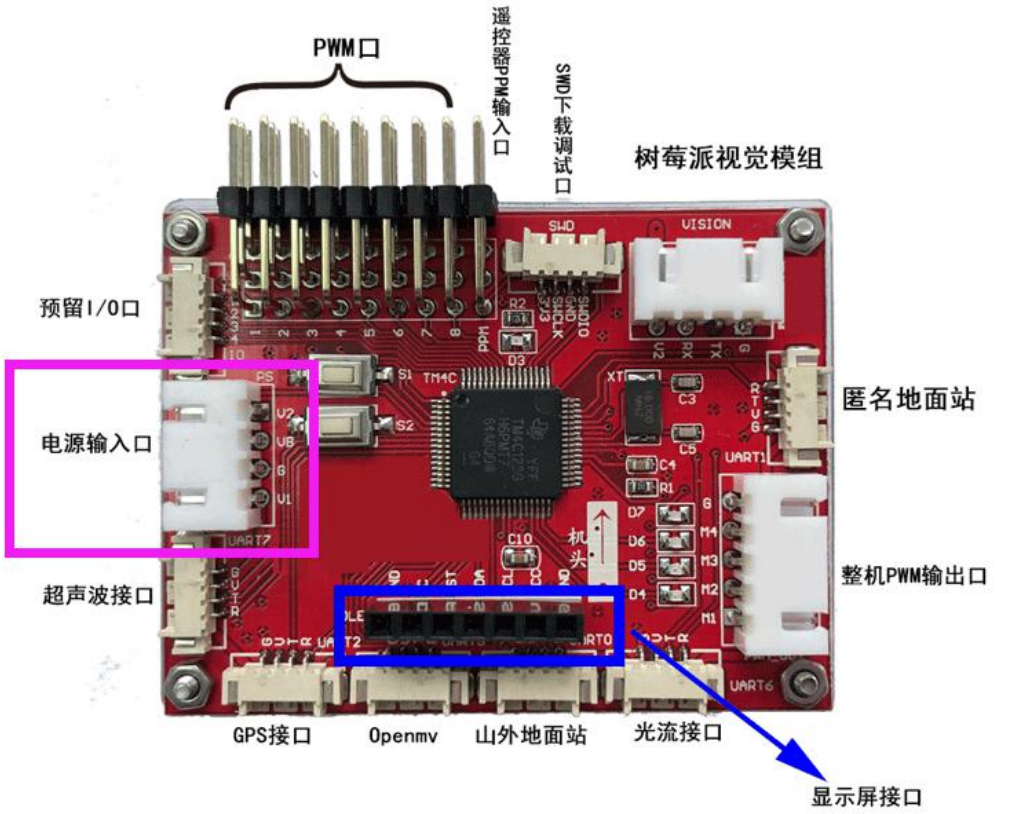
4.3 外设接口

2 路 I2C 接口、8 路 PWM 输出口、接收机 PPM 接口、OLED 显示屏接口、外部电压检测接口、1 个超声波接口、6 路通讯串口（接 OPENMV、GPS、光流、数传等）、2 个独立按键、1 个 SWD 下载调试口、4 个预留 IO 口、4 个可编程状态指示灯。

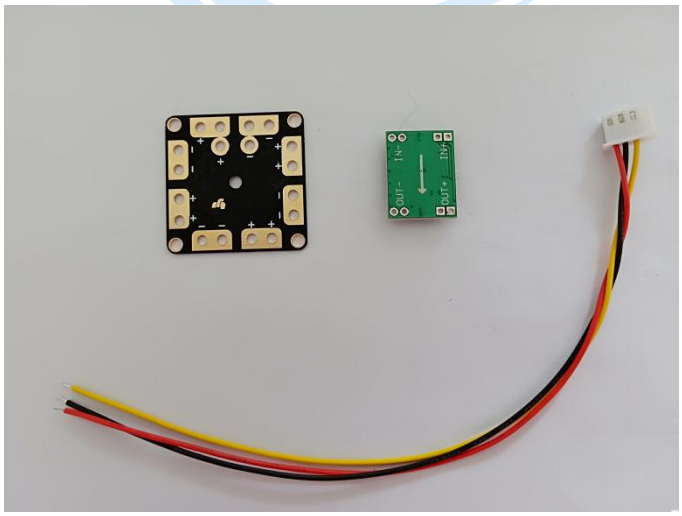


4.4 飞控主板总电源 5V 供电模式

飞控采用的供电方式为外部稳压模块输入 5V，接飞控电源输入口，注意电源正负级，V1 为正级 5V，G 为负级 0V。VB 为电池电压测量口，测量电池电压，飞控默认分压比位 11，故测量电压不要超过 3.3V*11=36.6V，若想测更大的电压，可以自行调整板子上分压电阻阻值。V2 为预留树莓派机器视觉模组单独 5V 供电口，因为树莓派功耗很大，和飞控公用一路 5V 3A 供电容易导致宕机，视觉部分不采用树莓派开发或者视觉部分本身有单独供电时，V2 可以不接。

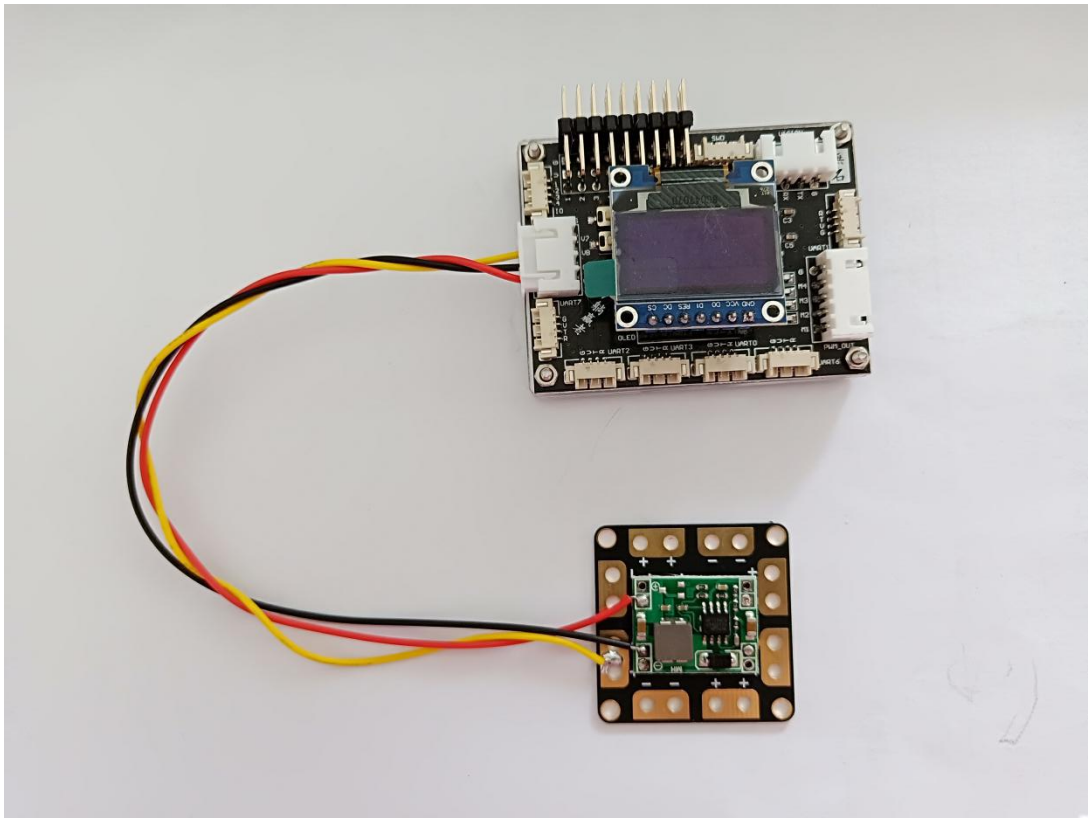


下面给出无名创新标准 NC330 样机的供电方式（以下模块不是飞控标配，需要单独购买），分电板+稳压模块+连接线，有的沉金版本机架自带分电功能，可以不需要分电板。

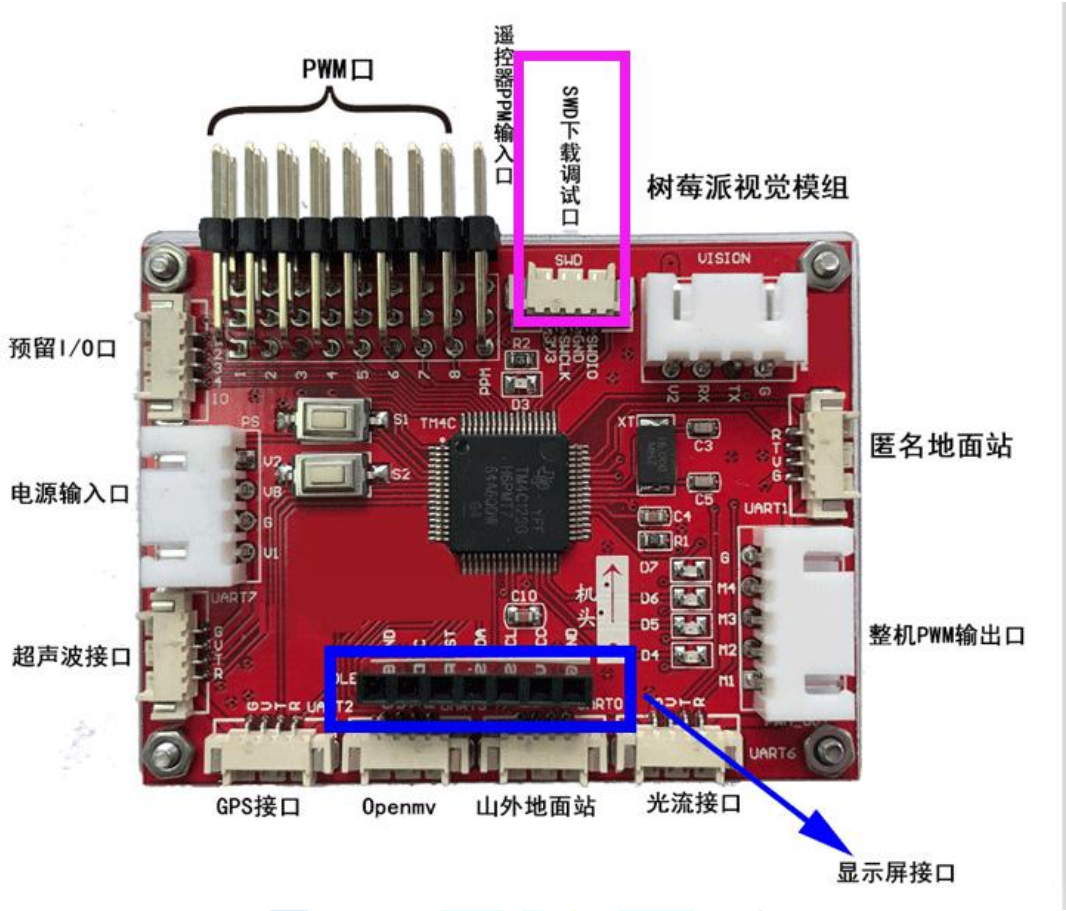


焊接时务必注意电源顺序，其中分电板上标出的+表示连接航模电池正极，分电板上标出的-表示连接航模电池负极，稳压模块上箭头方向表示为 DC-DC 稳压模块输入（航模电池）-->输出（飞控电源输入）的方向，即 IN+、IN-分别表示输入电压的正极和负极，

IN+、IN-分别接分电板的+、-。OUT+、OUT-分别表示输出电压的正极和负极。需要注意一点，此类 5V 3A 模块有两款。一种是不带电位器的恒定输出电压 5V，一种是带电位器的输出电压可调，使用是务必注意，为了确保安全，请在稳压模块输出接飞控前，用万用表测量电压是否正常。



其他电源接口：板上标的 V 统一表示 5V、3V3 表示 3.3V，其中只有 SWD 下载调试口处电源为 3.3V，**使用下载器连接时不要把 SWD 口的 3.3V 接成了 5V，TIVA 单片机对 5V 电压基本上是零容忍，很容易烧毁芯片**。



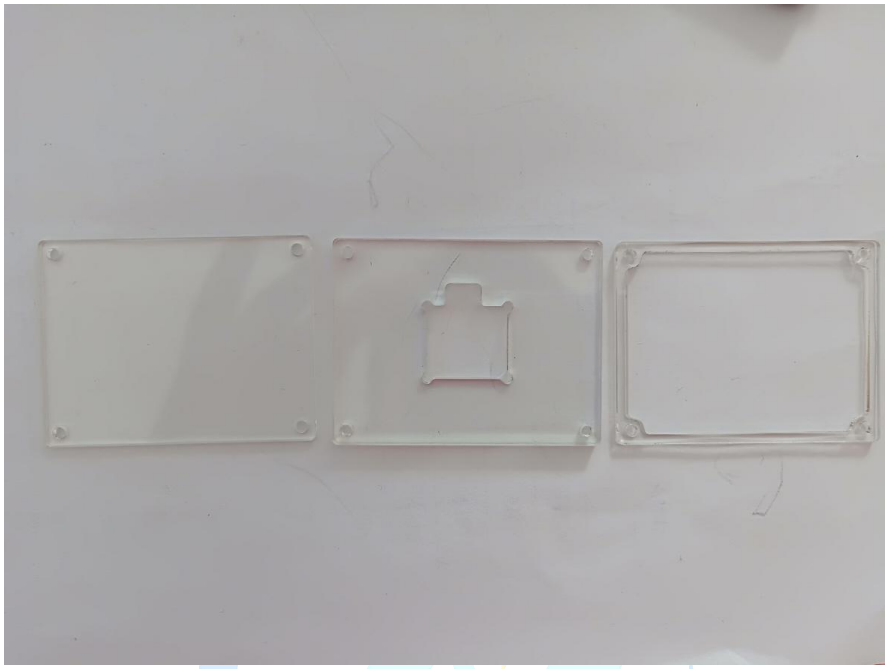
按键：S1 表示显示屏上一页，S2 翻页下一页、作用是结合显示屏查看飞控实时数据，显示屏常用页面显示内容见下文该部分说明。

LED 状态指示灯：其中 D5、D6、D7 为传感器校准、上锁、解锁等显示状态位，D4 位程序工作指示灯，D4 闪烁频率和当前 GPS 星数有关，星数越高闪烁频率会越慢，其中长亮表示 GPS 数据初始融合时刻出现较大波动值，建议快速重启后再起飞（针对飞 GPS 功能）。

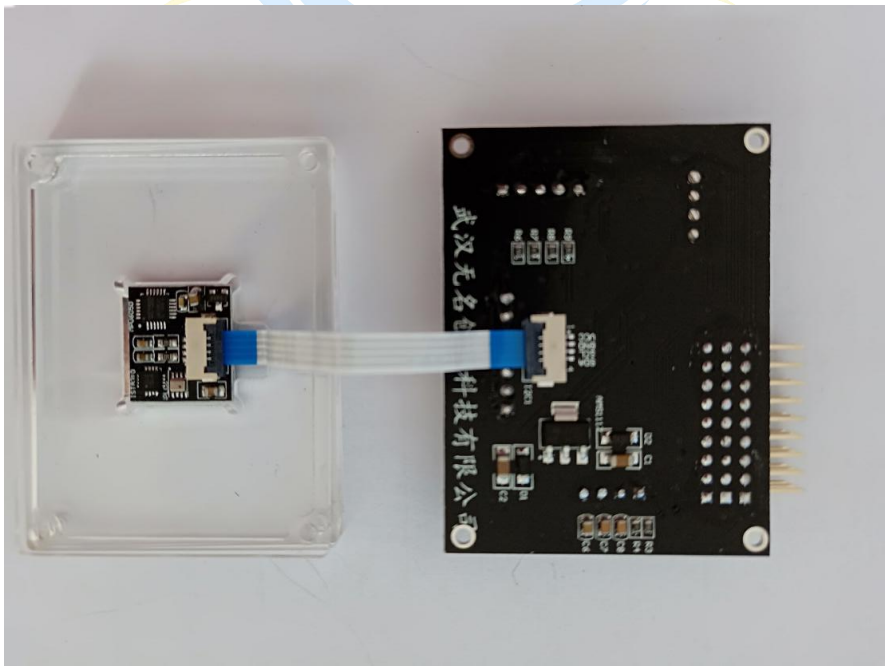
遥控器接收机输入接口：飞控解析遥控器数据采用 PPM 解析的形式，目前多旋翼飞控遥控器解析大多采用 PPM/SBUS 协议，时代在发展，行业在进步，PWM 接收方式，主要是为了方便早期航模玩家，直驱电调、舵机，现在主流多旋翼飞控均采用 SBUS/PPM 的形式，请放弃执念，与时俱进。新手经常抱怨自己遥控器说明书看懂不懂，介绍不详细，不会设置。天地飞、乐迪、福斯这类国产控，菜单界面大多都支持中文，自行摸索即可，几分钟就能解决问题，学习遥控器设置操作过程，远没有玩一款新手机 APP、一款新游戏复杂，多一点耐心即可。

五、飞控组装教程

将飞控亚克力外壳包打开，有如下 3 种类型的亚克力板子，从左到右依次编号 1、2、3，亚克力板子使用之前需要撕去外部保护膜。

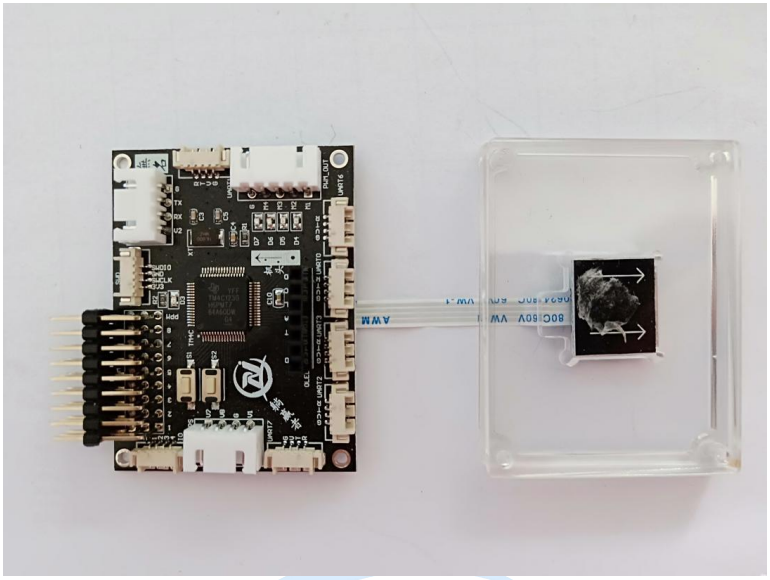


安装过程中 123 号板子，在飞控上的分布依次从下到上，首先将 1、2 板子用长螺丝从底部穿入，同时将传感器端用软排线连接，FPC 排座为翻盖式，使用时需要用手拨动黑色保险盖，完成后如下图所示，注意排线蓝色所在方向需要和下图一致。

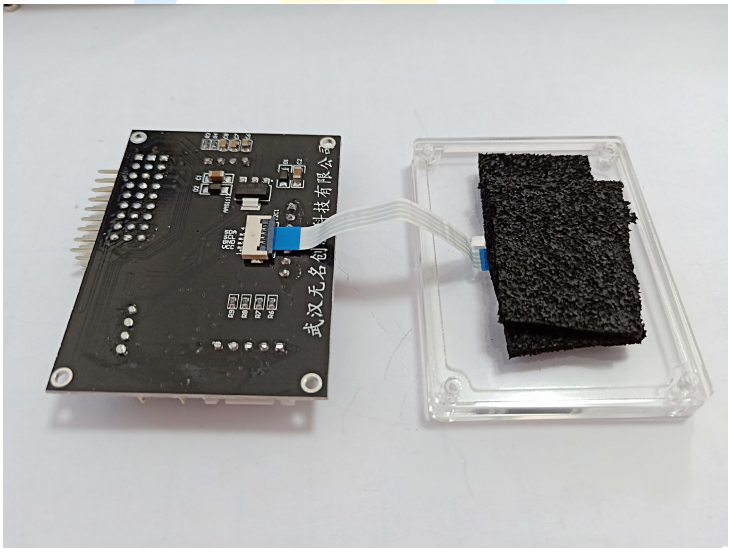


将传感器背面粘一段双面胶如下图所示，然后粘贴在传感器槽内，注意尽量顶着槽

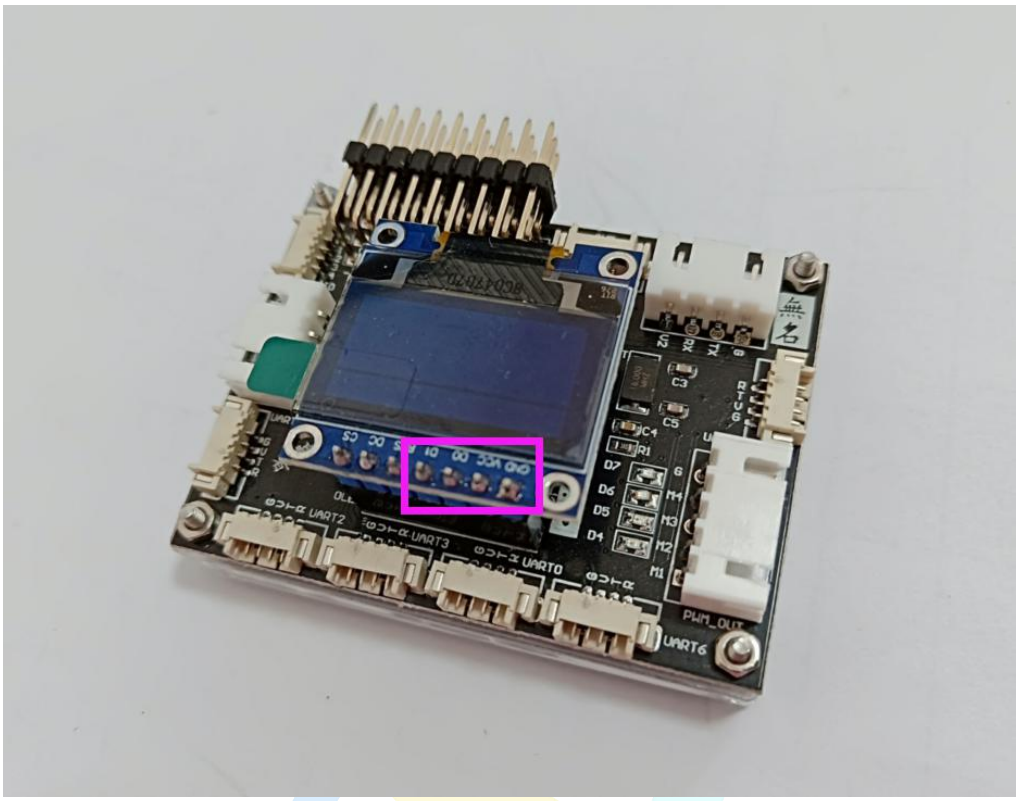
的边安装，这样避免安装造成传感器外部额外的正交化误差，**传感器箭头方向即为机头方向**。



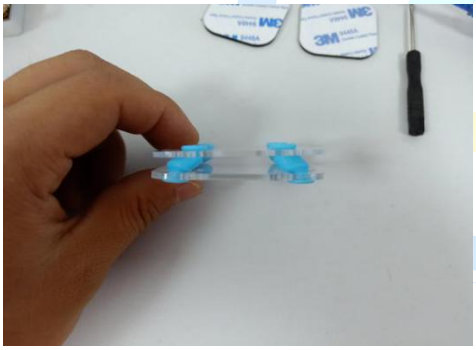
软排线另外一端连接飞控板底部的 I2C1 排座口，在 3 号板子槽内亦可塞入一定量海绵，完成后如下图所示。

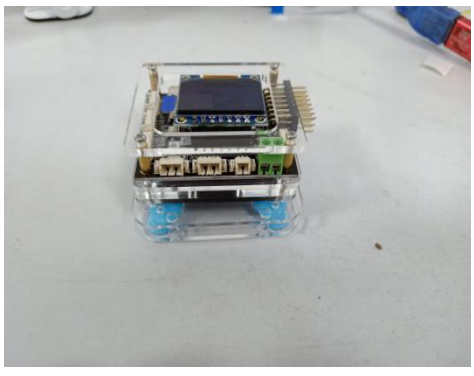


将主板放置于底部传感器缓冲盒上，并拧号铜柱完成，将 5 号板子用小号螺丝拧在铜柱上，最后插入显示屏如下图，对于 4 针的 OLED 只插前四个槽（下图标红处），减震架与飞控板间用 3M 胶粘住即可。



斜拉飞控减震架安装如下图：



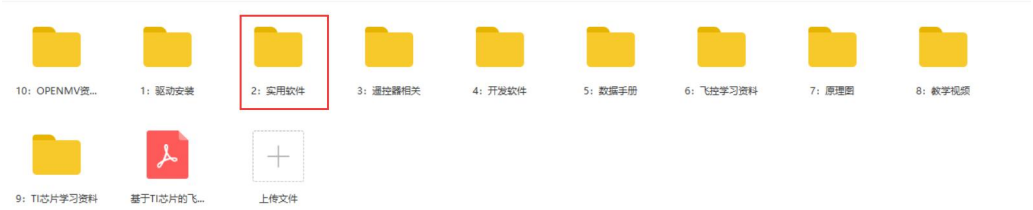


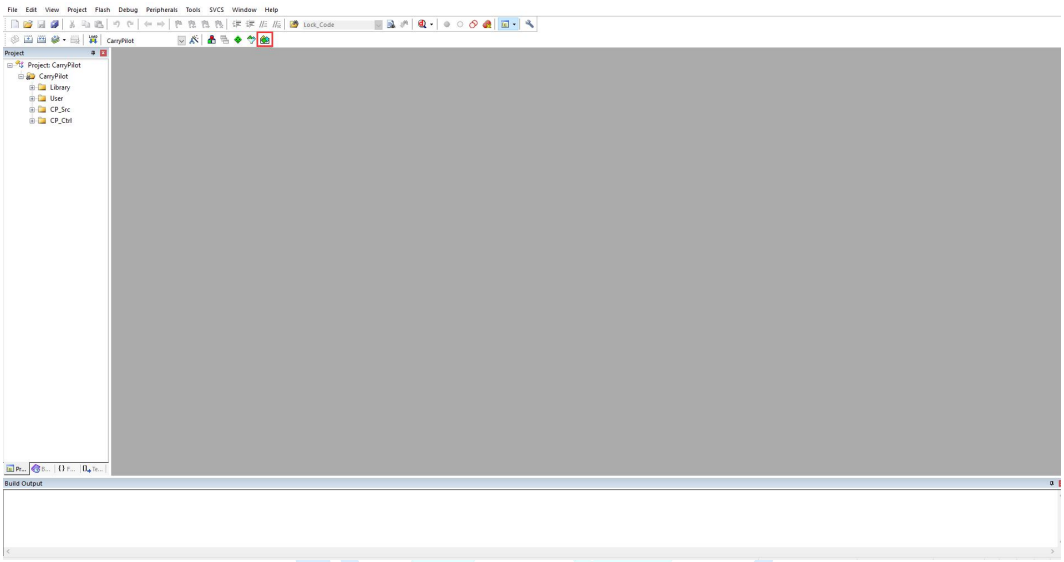
六、编译环境的搭建

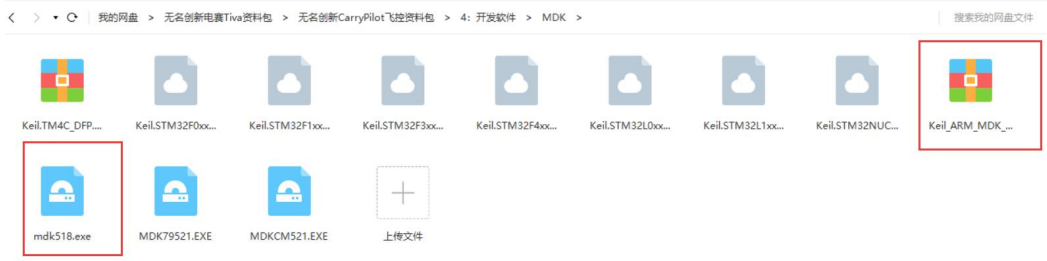
6.1 MDK(Keil)开发环境的搭建

1、按正常流程安装并破解 MDK5.25 版本，资料包下有安装包与破解文件，其它支持包根据自己需求选装。

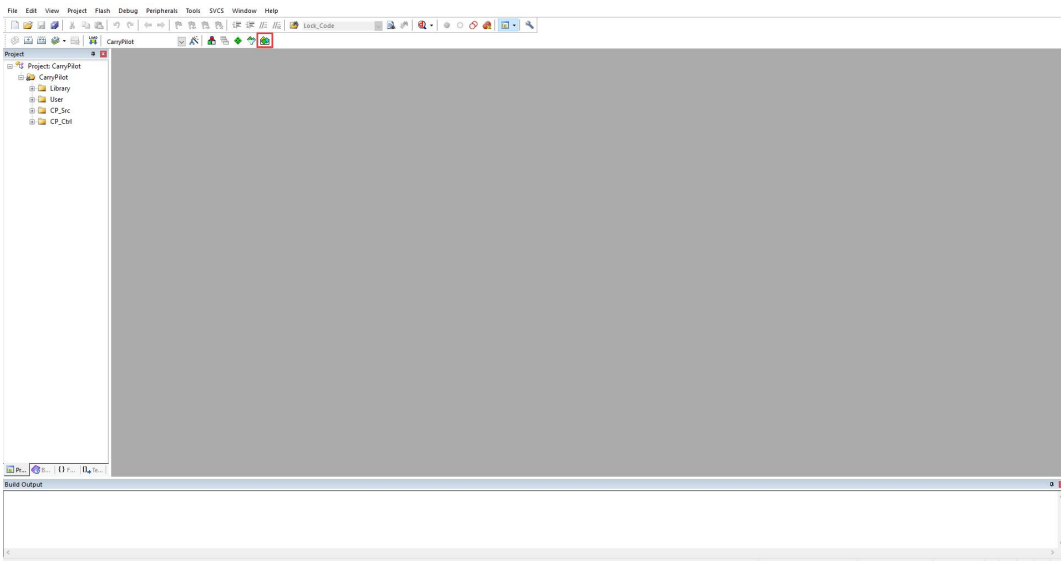
https://jingyan.baidu.com/article/2f9b480d9ecf9f41ca6cc248.html



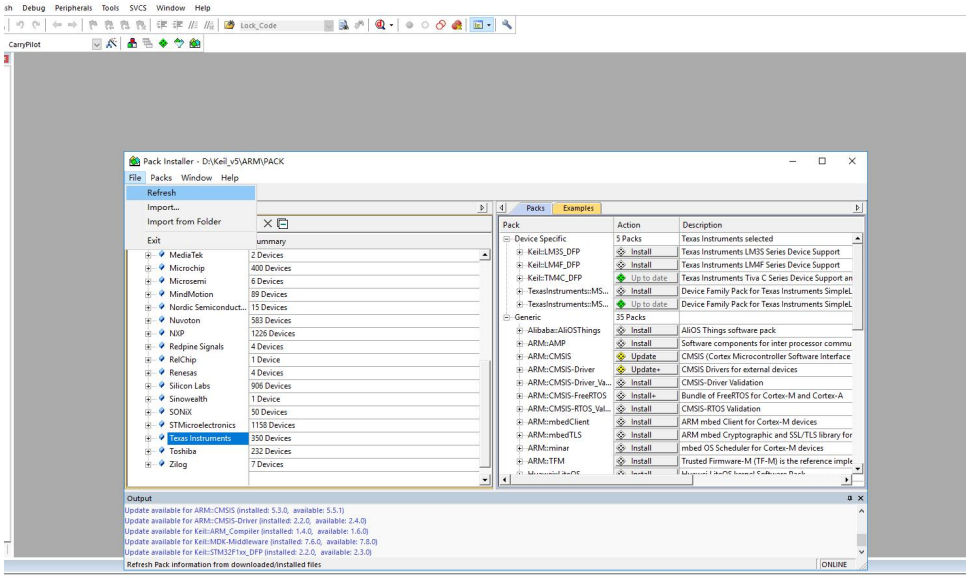


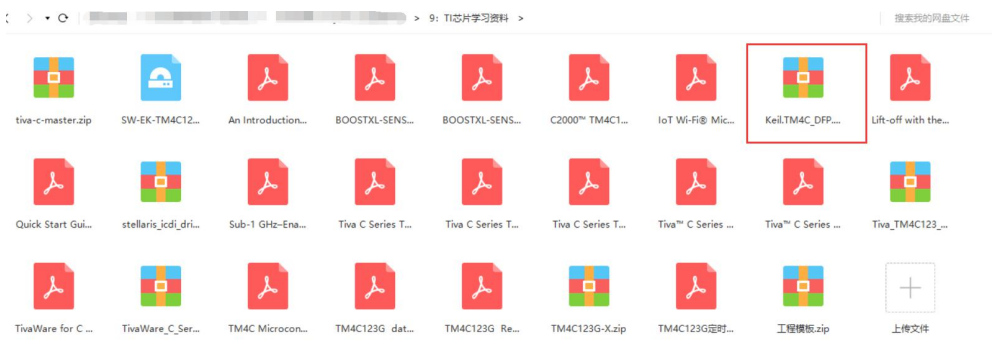


2、安装破解后，进行离线更新 TIVA 支持包，点击 Pack Installer，进入支持包更新界面。



3、点击 File-Import，找到资料包下，找到 Tiva 离线支持包导入





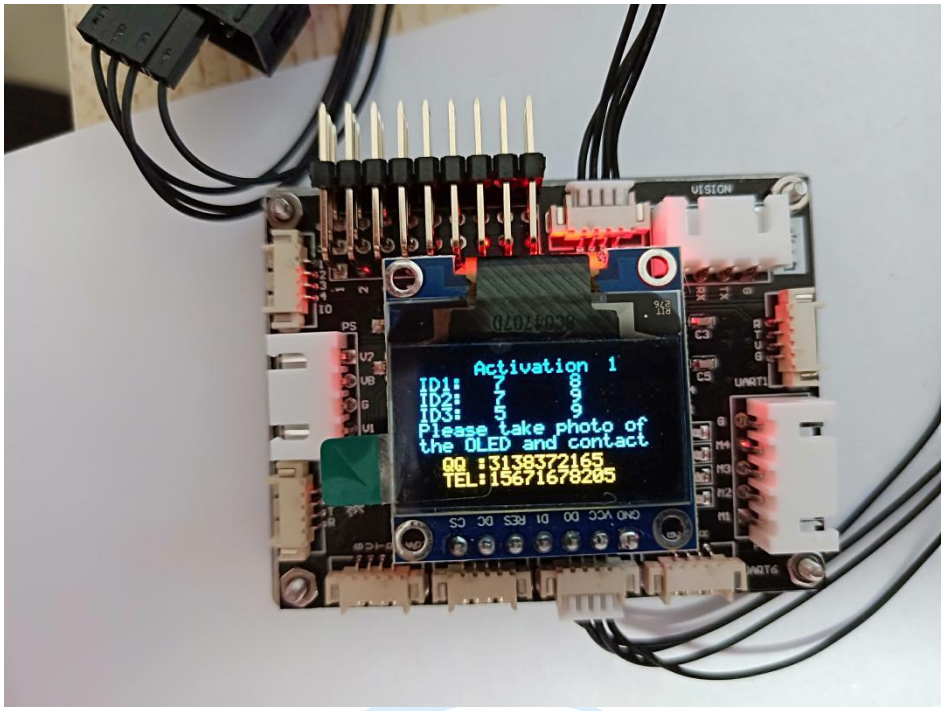
4、IAR 开发环境环境的搭建，按照资料包破解视频教程一步一步来即可。

七、飞控正版激活操作

为了使得无名飞控事业的走可持续发展路线，激励初创团队更好地为广大飞控爱好者提供教学视频、技术博客、售后技术服务等，我们鼓励大家支持原创、正版，因此对程序里面一部分内容做了适当加密保护。开源并不等于免费，前辈们的历史已经证明，在国内这个环境下，毫无收益的开源，单靠坊间个人爱好者，自发地参与项目完善的方式行不通，好的开源项目需要请专职人员做好售后服务、手把手教学、统计用户反馈需求、在实践中完成对产品的一次次完善与迭代升级。

用户如果自己更换过单片机、误写程序擦除全部 FLASH 与 EEPROM，再次使用飞控前，必须经过激活操作。飞控在出厂测试时，已对飞控进行激活操作，只要用户不对飞控进行擦出芯片全部 FLASH 与 EEPROM 操作，就不必进行重新激活，（首次无需进行激活操作）用户拿到飞控后，用户首次拿到的飞控板里面是出厂激活的程序，无实际功能，需要自行编译工程代码、使用下载器烧入飞控即可。

误擦全部 Flash 后激活操作，开机上电会进入如下界面提示需要进入飞控正版激活操作。



飞控串口 0 接山外调试助手时，提示界面如下图：

客户自己设计飞控时，若使用无名飞控代码，需要激活后才能使用，激活只对无名创新客户开放，单次激活 98 元，因芯片烧毁从新更换单片机、自己擦除全部 FLASH 与 EEPROM 操作的情况，同样以此标准。

通过提供的联系方式加 QQ：3138372165，支付激活费用并获取激活码后，通过山外地面站连接主串口后，将激活码发送至飞控端即可实现激活。山外地面站端选用串口调试助手子窗口，波特率配置为 115200,8 位、无奇偶校验、停止 1 位 1。发送时以字符

（非十六进制）形式发送。

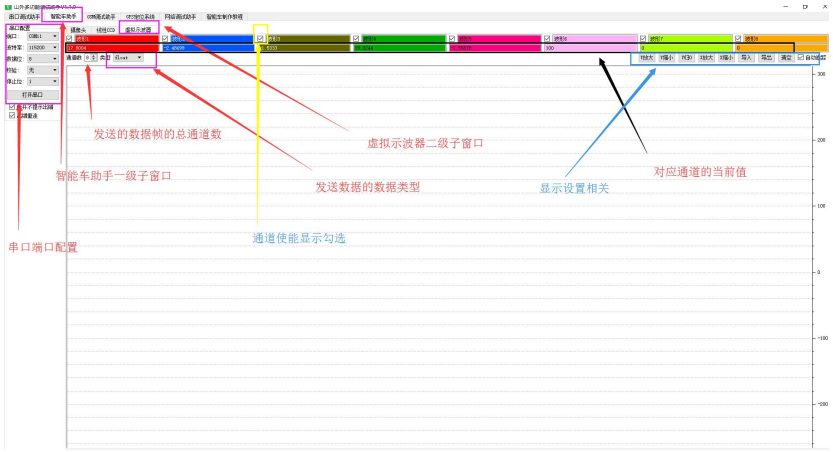
激活成功后，飞控端会返回提示如下图，只要下载不勾选全部擦除全部 FLASH 与 EEPROM，飞控后续就无需进行激活操作。

八、地面站使用教程

无名飞控可以支持多家上位机：山外多功能调试组手（连主串口 0）、匿名上位机（连主串口 1），其中山外上位机主要用作观看飞控数据实时波形，匿名上位机主要用做观察飞控状态、更改飞控控制参数等。两个上位机可同时使用。

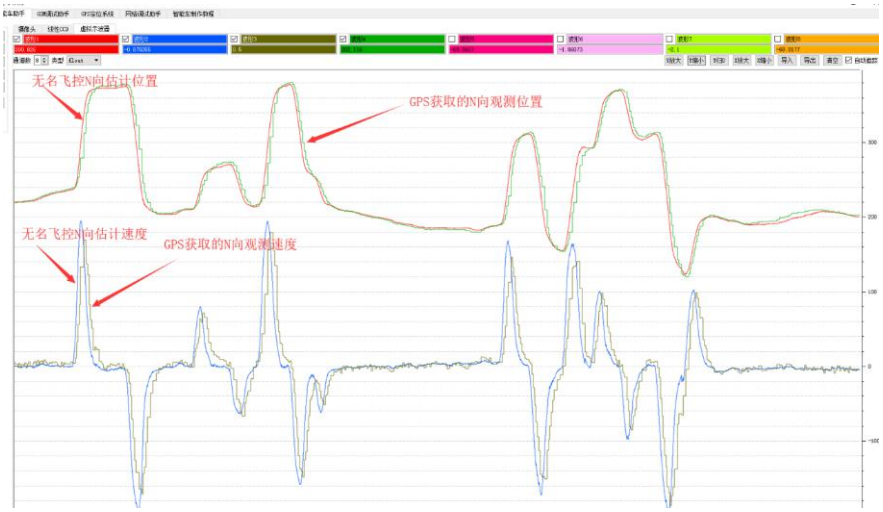
8.1 多功能调试助手的使用

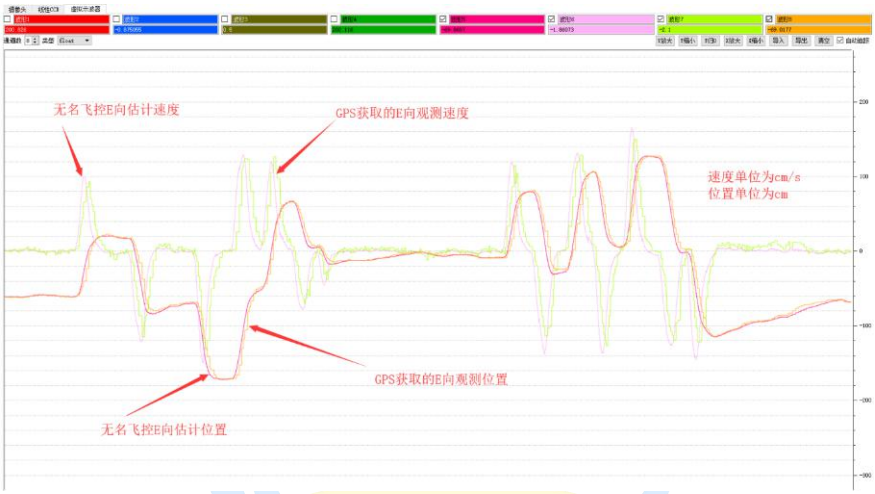
无名飞控连接山外上位机最常用的是数据波形显示功能，初始界面如下图。



在使用山外多功能调试助手显示数据实时波形时，飞控端是连主串口（UART0），电脑端打开软件后，注意是在：智能车调试助手-虚拟示波器子窗口下，需要对端口号（查看电脑设备管理器确定）、波特率（115200）、通道数（默认 8 通道）、数据类型（勾选 float）即可显示对应通道内数据波形如下。

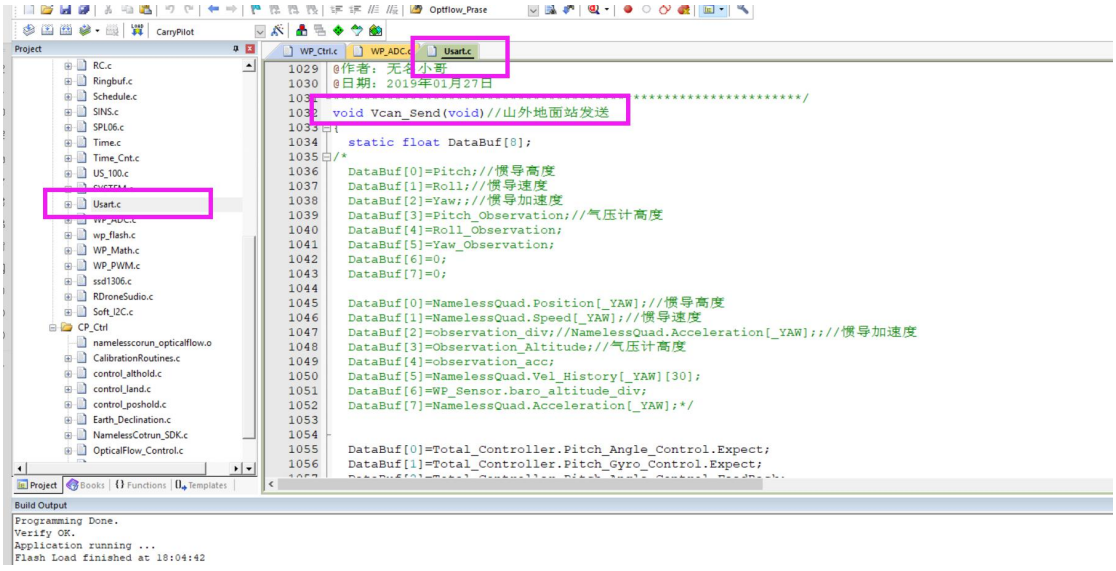
高度方向惯导融合图





水平方向惯导融合图

使用山外上位机时，如果想改对应通道数据，可在程序 USART.c 函数里面 Vcan_Send 函数对应通道赋想要查看的变量即可。



更多山外上位机使用说明见使用说明书，山外论坛网址：

http://www.vcan123.com/forum.php

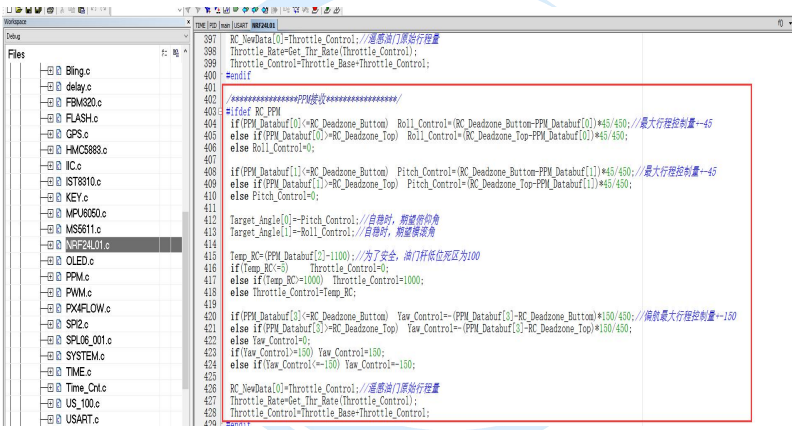
8.2 匿名上位机的使用

无名飞控连接匿名上位机主要作为观察飞控相关状态，比如：传感器原始数据、姿态角、高度、GPS 位置、遥控器数据、PID 参数等功能，也可以显示数据波形，但是不推荐使用匿名上位机看数据波形。串口配置需选择对应串口号、手动输入波特率任意，最后点击右下角的打开连接即可。观察飞控状态勾选左侧-飞控状态子窗口，即可显示飞控相关状态如下图。

使用时，可以前后转动、抬高飞机观察上位机数据变化，当 GPS 定位成功时，观察

星数、位置坐标，接上遥控器后，可观察遥控器：对前 4 个通道（1、横滚；2、俯仰；3 油门；4、偏航）进行动作与通道数据变化一一对应，本飞控默认采用左手油门（美国手），前四个通道对应动作位如下图（标准美国手的默认方式）。

实际操作中，若发现 1~4 通道动作不是分别对应横滚 ROL、俯仰 PIT、油门 THR、偏航 YAW 数据变化，可以通过设置遥控器的 Sticks Mode 来实现一一对应，若遥控器不支持更改，也可通过该程序里面对应通道赋值也可以实现，见遥控器接收 RC_PPM 代码段。



确定好前 4 个通道对应关系后，观察遥控器动作位（横滚、偏航左右动作；俯仰、油门上下动作）增减变化是否与标准 APM、Pixhawk 规定的是否一致。

横滚杆从中位向右侧打杆对应横滚通道数字量递增，横滚杆从中位向左侧打杆对应横滚通道数字量递减。（从左到右，数字量递增）

俯仰杆从中位向下侧打杆对应俯仰通道数字量递增，俯仰杆从中位向上侧打杆对应俯仰通道数字量递减。（从上到下，数字量递增）

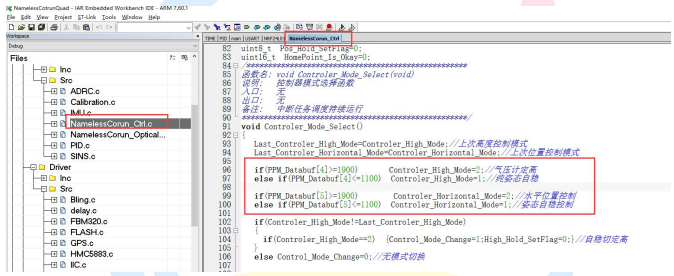
油门杆从中位向下侧打杆对应油门通道数字量递减，油门杆从中位向上侧打杆对应油门通道数字量递增。（从下到上，数字量递增，与俯仰杆动作位变化相反）

偏航杆从中位向右侧打杆对应偏航通道数字量递增，偏航杆从中位向左侧打杆对应偏航通道数字量递减。（从左到右，数字量递增，与横滚角动作位变化相同）

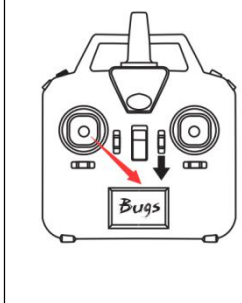
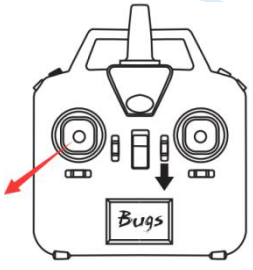
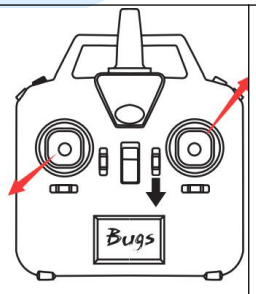
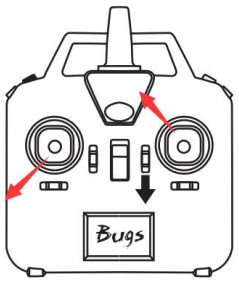
实际操作过程中，若发现哪一个通道与上述变化不一致，需要设置遥控器对应通道

反向即可。确定好上述步骤后，还需要确定下遥控器回中时，横滚、俯仰、偏航通道的中位值，标准 Futaba 中位为 1516 左右，如若发现自己的遥控器中位不在这个值附近，可以通过遥控器上面的微调按钮来微调，因为飞控默认的中位死区为 100。如果遥控器实际偏离标准中位太远（超过 50），不微调会导致解锁、上锁、校准失败，或者起飞会一直往一边偏飞的情况。实际如果观察某次中位在 1516 附近，可以尝试再次动作自然会中后，再次确认下，确保因遥控器质量、使用寿命等导致电位器回中拨动范围在合理范围内。

确定好前四个通道后，对遥控器辅助模式档位进行确认，观察开关在低位的数字是否小于等于 1100，开关在高位的值是否大于等于 1900（通常高位为 1000、低位为 2000），如若发现自己打摇动器低位比 1100 大，有两种办法：1、设置摇动器该通道端点值；2、对应把飞控模式函数的幅值条件改至瞒住即可，比如发现自己遥控低位为 1150，需要把低位幅值条件改为 1160 即可，当高位不满足时操作同理。



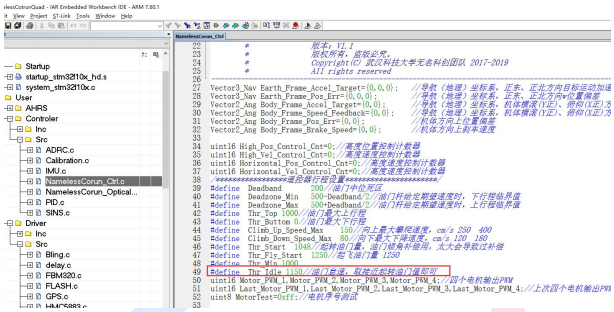
结合地面站确定好上述遥控器设置后，即可对飞控进行解锁、上锁、传感器校准操作。

			
解锁	**上锁**	**加速度计校准**	**磁力计校准**

8.3 解锁

边遥杆处于右下角，持续 1.5S 左右，飞控会进入解锁状态，当解锁后，D5、D6、D7 指示灯会进行流水灯慢闪状态，解锁后 6 秒钟内，如果无推油、打杆动作，飞控会

自动上锁，同时 D5、D6、D7 指示灯会同步快闪 3S。若解锁后 6S 内，有推油、打杆动作，飞控会一直处于解锁状态同时 D5、D6、D7 指示灯会进行流水灯快闪状态 3S。飞控处于解锁状态时如果满足着地条件，电机会进入怠速状态（以较低的油门量转动）；当不满足着地条件时，飞控油门输出来源于实际控制器的输出。若新手觉得怠速不安全，可以将程序里面怠速值调小或者直接给最低位 1000。如果发现解锁后怠速太小，可以设置电调行程或者改程序怠速值，实际为了安全，怠速值不宜太大，基本接近实际电调起转油门就可以。



8.4 上锁

左边遥杆处于左下角，持续 1.5S 左右，飞控会进入上锁状态。

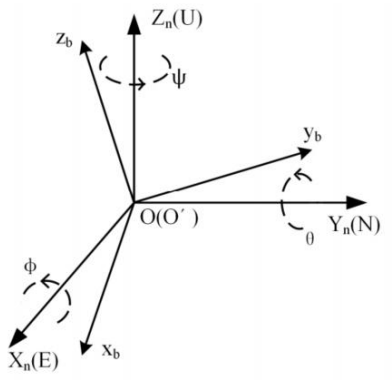
8.5 校准加速度计

左边遥杆处于左下角，右摇杆处于右上角，持续 3S 左右进入，进入后需把飞机分别放置于 6 面，某一面放置好后，通过遥控器动作位，指定该面进行采集，加速度计 6 面校准时，采集某一面数据时应尽可能保证水平放置。进行某第 N 面采集时，D3、D4、D5 指示灯会对应显示竖字量 N（默认 1~6 顺序依次是正面、左测面、右测面、前测面、后测面、下面），该面校准完毕后（6 面数据未全部采集），会进入待机闪烁，此时将继续放置于下一面，在通过遥控器指定该面即可，待 6 面数据都采集好后，即可完整校准。

```
/*第一面飞控平放，Z轴正向朝着正上方，Z axis is about 1g,X、Y is about 0g*/
/*第二面飞控平放，X轴正向朝着正上方，X axis is about 1g,Y、Z is about 0g*/
/*第三面飞控平放，X轴正向朝着正下方，X axis is about -1g,Y、Z is about 0g*/
/*第四面飞控平放，Y轴正向朝着正下方，Y axis is about -1g,X、Z is about 0g*/
/*第五面飞控平放，Y轴正向朝着正上方，Y axis is about 1g,X、Z is about 0g*/
/*第六面飞控平放，Z轴正向朝着正下方，Z axis is about -1g,X、Y is about 0g*/
```

加速度计 6 面校准无先后顺序，只需确保所放置面与遥控器指定采集的数据面一致即可，推荐按照默认顺序采集，免得忘记顺序导致出错。

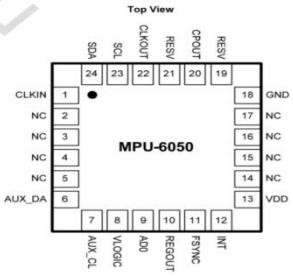
无名飞控载体系 b 与导航系 n 坐标系统定义：



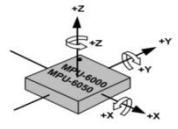
导航坐标系与载体坐标系的关系

选取“东北天”（ENU）为导航坐标系(n系)，右手笛卡尔直角坐标系为载体坐标系(b系)，设初始时刻，这两种坐标系重合，在建立上述坐标系之后，为了描述刚体在三维欧几里得空间的取向，通常采取欧拉角来定义三轴姿态角度，图中绕 XYZ 轴分别对应的名称为倾仰角（Pitch，ϕ）、翻滚角（Roll，θ）和偏航角（Yaw，ψ）。

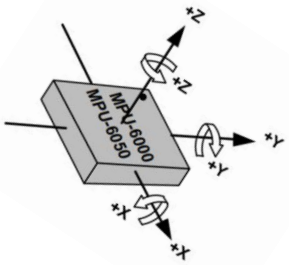
定义上述坐标系后，在传感器安装时，默认安装是将 MPU6050 的三轴加速度 XYZ 与坐标系载体系统的 XYZ 进行同轴、同向安装，即：绕着加速度计的 X 轴逆时针旋转为 Pitch 角的正方向，绕着加速度计的 Y 轴逆时针旋转为 Roll 角的正方向，绕着加速度计的 Z 轴逆时针旋转为 Yaw 角的正方向，即加速度计传感器 Y 轴正向对应飞控机头正向。

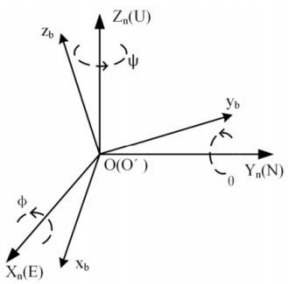


QFN Package
24-pin, 4mm x 4mm x 0.9mm



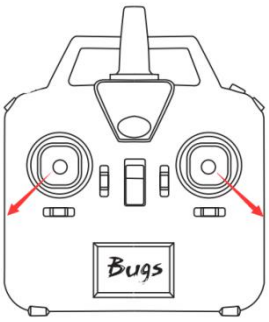
Orientation of Axes of Sensitivity and Polarity of Rotation





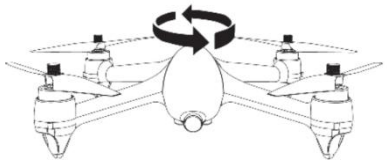
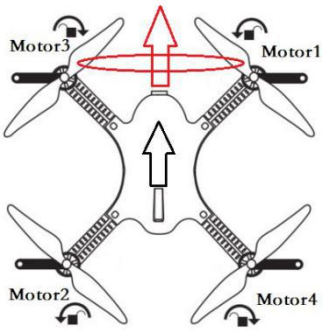
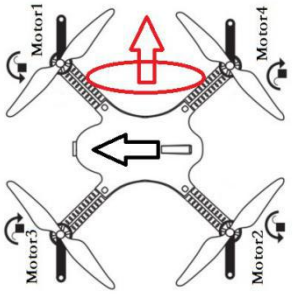
8.6 机架校准水平

飞控安装在机架后，连同机架一起校准好加速度计后，将飞机放置于水平地面，重新给飞控上电，持续外八动作位 10s 左右，待状态指示灯快闪后，即完成机架水平校准，校准动作位如下图。（校准水平前需先校准加速度计）



8.7 校准磁力计

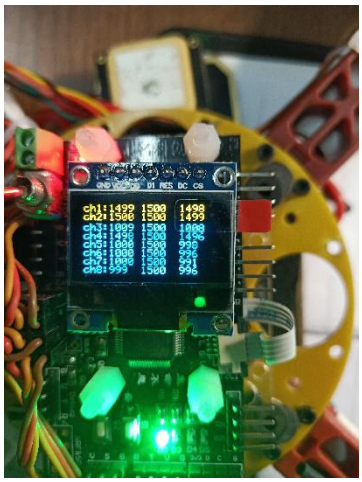
左边遥杆处于左下角，右摇杆处于左上角，持续 5S 左右进入，需让飞机沿着水平面、机头朝上面转 360 度完成采集，进入后需指定先采集水平还是机头右侧朝上，通过遥控器左侧遥杆设置，其中左侧摇杆处于右下表示进行水平面采集，水平转 360，该面采集完毕会自动进入机头朝上面采集，需把飞机机头朝上，旋转 360 后即可采集数据。在旋转过程中，可能会出现转一圈后，某一面仍然为采集好，需重复转圈至所有角点采集完毕，旋转是请尽量保证水平面旋转时水平、机头朝上时竖直、机头右侧朝上时竖直。（图示中，红色箭头指的方向为天，机身上的箭头为机头方向）

		
水平旋转 360 度	**机头朝上旋转 360 度**	**机头右侧朝上旋转 360 度**

8.8 遥控器行程校准

给飞控上电的同时，长按住飞控上面按键 S1，待飞控显示屏出现“MS Is Okay”时松开按键 S1，飞控会自动进入遥控器行程校准界面。OLED 显示界面如下图所示，默认

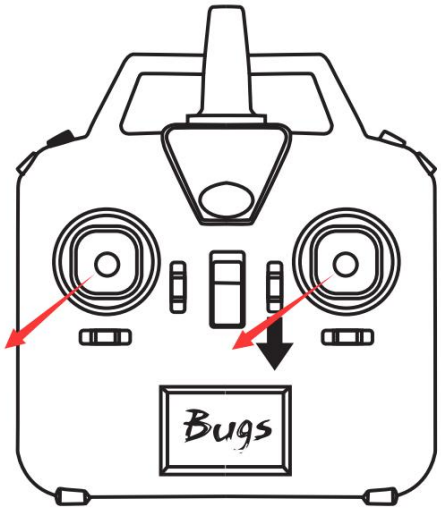
值全部写 1500。



在此界面下，操作遥控器，使得各个通道最大、最小行程均被采集到，确保每个通道都有动作后，按下飞控按键 S2 松开后完成遥控器行程校准。**（提示结束时必须按下 S2，否则该次校准无效）。**

8.9 电调校准

卸掉电机上的螺旋桨（一定要切记，避免血肉模糊），上锁状态下，左边遥杆处于左下角，右摇杆处于左下角，持续 5S 左右进入，此时飞控会进入电调校准准备模式，显示屏会提示先拔掉电调总电源，并将遥控器油门杆置于最高位。进入动作位与进入后提示界面如下图所示。



完成上述操作后，（卸掉电机上的螺旋桨,一定要切记，避免血肉模糊），需重新插上总电源，注意插的时候不能抖动，避免反复给主板供电后，导致电调行程校准失败。对于第一代飞板控，可在拔掉电池时先将主板供电开关切到断开状态，再插上电池电源

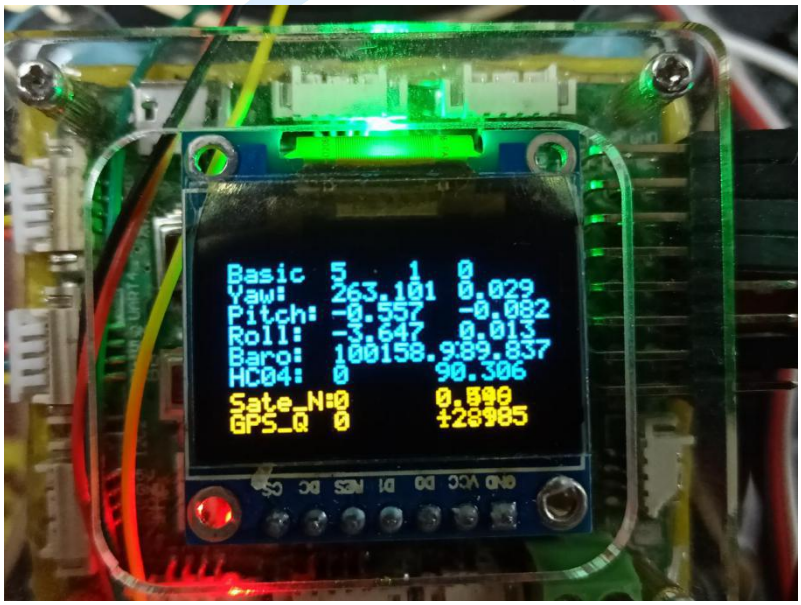
后，再给主板供电，再次供电时，遥控器油门杆应保持最高位，等待电调进入行程校准模式时（每个电调声音不一样，需阅读说明书，通常时“哔哔哔”时表示行程校准），迅速下拉油门杆于最低位，待电调发出正常低位待机声音后（通常是发出“do re mi”），表示电调校准完毕。校准完毕后可以推油确认下是否校准 Okay。

（对上述解锁、上锁、校准操作介绍不详尽的地方，请结合配套讲解视频学习）

九、OLED 显示屏主要页面介绍

飞控板载 OLED 在使用过程中用于实时观察飞控内部运行状态、关键数据等，使用好显示屏可以替代地面站的一部分工作，更高效的开发和飞行。下面对显示屏主要页面做介绍。

(1) 第一页：基础数据显示页



第一行的第一个数据为主定时器系统调度时间，正常状态为 5ms，用户在增加、修改任务时，系统调度可能会超时，就会导致对周期性要求极高的：传感器采集、姿态解算、惯性导航、控制等任务得不到保障，飞控会出现异常。所以在修改代码后，可以观察此值是否严格在 5ms 附近。由于本飞控整个任务调度、优先级经过严格时间设计，目前飞控单次最大任务调度时长接近 4ms，最低 2.1ms，设计成 5ms 任务调度为了保障 GPS 用于有足够的时间开销去二次开发。第一行的第二个数据为显示屏页数，当前显示 1 表示处于第一页，也是开机后默认进入的页面。第一行的第三个数据为外部电压指示，外部分压电阻是 10：1，最大测量电压可以到 33V，常用的 2S-6S 电池都可以测量。

第二、三、四行的第一个数据表示姿态角度（单位为 deg），后一个数据表示姿态角速度（单位为 deg/s）。开机上电时请不要晃动无人机，初始化时，飞控会自动采集陀螺仪当前温度下零偏，若出现晃动，陀螺仪初始零偏采集错误，姿态解算基本就是崩盘的。

第五行的第一个数据为气压计获取的压强值，第二个数据为当前气压与初始地面气压通过“压差法”计算得到的相对高度值。

第六行的第一个数据为超声波得到的对地高度值，第二个数据为惯导融合得到的高度估计值，当超声波传感器有效时，优先采用超声波对地高度作为观测量修正高度方向上的惯导融合。

第七行的第一个数据为 GPS 定位星数，用于配合飞控工作状态指示灯，观察 GPS 定位是否 Okay。一般飞 GPS 模式时确保在地面就满足 GPS 定位模式，首次满 GPS 定位条件时，会刷新返航点，用于一键返航、失控返航时的目标点。第二个数据为高度方向上的惯导估计速度值。

第八行的第一个数据为 GPS 定位质量，值越小表示定位质量越好，通常值小于 2.0，第二个数据为数值方向上惯导原始加速度，此值校准好加速度计后，基本是带一定噪声值在 0 附近跳变，允许存在一定静差（正负 10 以内），若静差过大，会影响惯导估计的效果，所以当发生飞控安装位置变动、炸鸡后观察此值，若静差过大，请从新校准加速度计、机架水平后再起飞。

(2) 第二页：传感器校准数据显示页

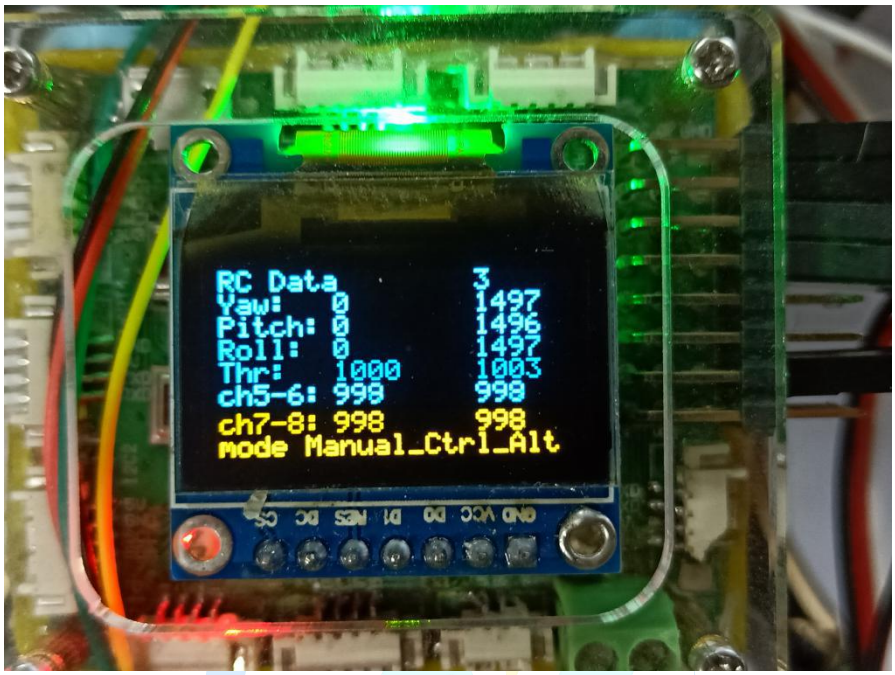
第一行的第一个数据为当前所在页面，2 表示当前处于第二页。

三、四行的第一个数据为加速度计六面校准后得到的量程校准参数，通常校准后的量程接近数值 1，第二个数据为加速度计校准得到的零偏量。

第五行的数据表示当前所处的磁场强度，正常的地磁强度为 0.5 左右，观察此值可以判断当前所处磁场是否存在较大干扰。

七、八行的数据表示磁力计经过椭球拟合后，得到的磁力计中心偏移动量。

(3) 第三页：遥控器控制量数据显示页



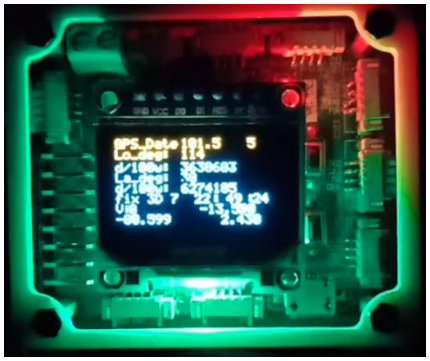
第一行的第一个数据为当前所在页面，3 表示当前处于第三页。

三、四、五行的第一个数据为表示经过运算转化得到的偏航、俯仰、横滚、油门控制量，右边摇杆处于自然回中状态、偏航处于中位、油门处于低位时，校准完毕后的遥控器控制量应该如上图所示，即偏航、俯仰、横滚控制量均为 0，油门控制量为 1000，若飞控解解、上锁、传感器校准等功能不正常，请观察此页来判断是否需要从新进行校准遥控器或者调节遥控器微调按钮操作。第二个数据为对应通道的原始数字量。

七行的数据分别表示遥控器 5、6、7、8 通道的原始数字量。

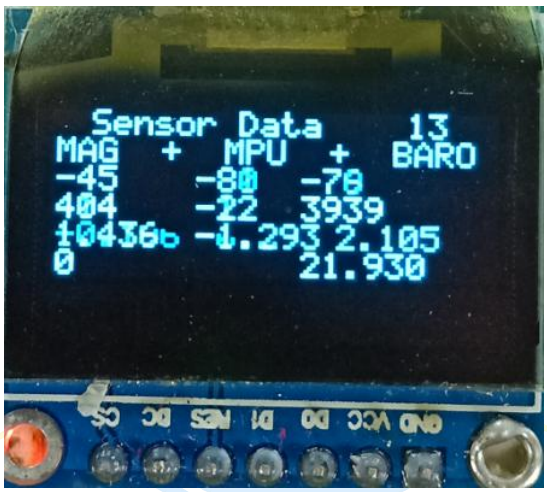
第八行字符串表示当前所处模式，用户使用时可以结合遥控器功能设置开关通道（见操作说明）观察是否进入对应模式。

(4) 第五页：GPS 定位数据显示页



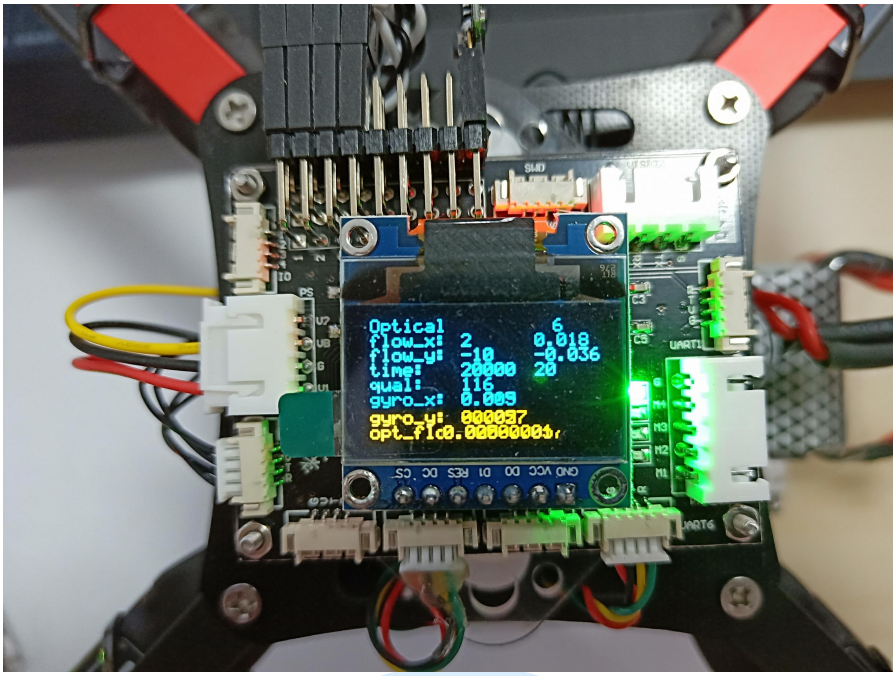
第一行第一个数据表示 GPS 解析周期，无名飞控 GPS 工作在波特率 921600、刷新频率 10Hz，解析 PVT 语句，飞控再整体设计时，已经确保了数据不丢帧，但是因为存在抢占式中断处理机制，本时间不是严格的 100ms，接收时间在 98~102ms 之间均属于正常。第二、三、四、五行表示 GPS 得到的定位经、纬度信息。第 6 行第一个数据表示定位状态（3D 定位 、2D、nofix）,第二个数据表示用于当前定位的微星数量，第三个数据为 GPS 获取的 UTC 时间数据（已转化为北京时间），第七行分别表示 GPS 获取的东向、北向速度信息。第八行第一个数据表是 GPS 获取的天向速度信息，第二个数据表示当前 GPS 定位质量。**GPS 安装时请保持陶瓷天线朝上、上方无金属物件遮挡**。在 GPS 首次定位成功时，只会输出时间数据，其它数据均为 0，若长时间等待仍未有效定位，尝试换到空旷环境飞行。

(5) 第十三页：传感器原始数据显示页



第二、四、五行的依次表示磁力计、加速度计、陀螺仪的 X、Y、Z 三个方向的原始数字量，第六行为气压计数据，第七行为气压计得到的高度观测量。

(6) **第六页：光流数据显示页**



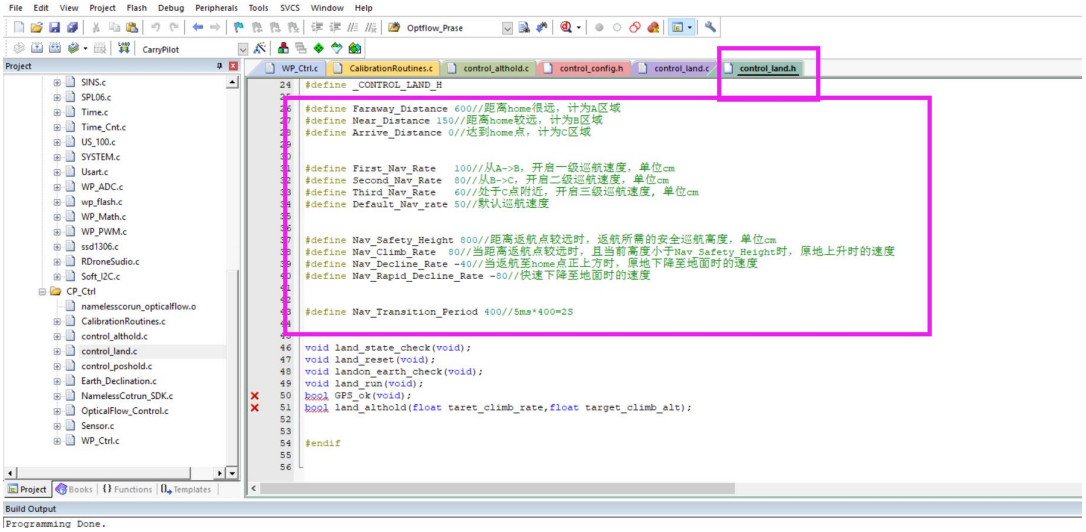
第二、三行第一个数据表示原始 X、Y 方向的像素偏移，第四行为测更周期，第五行为定位质量。第六、七行为 X、Y 方向的光流角速度。第八行为滤波后的光流角速度。

十、结合地面站调参

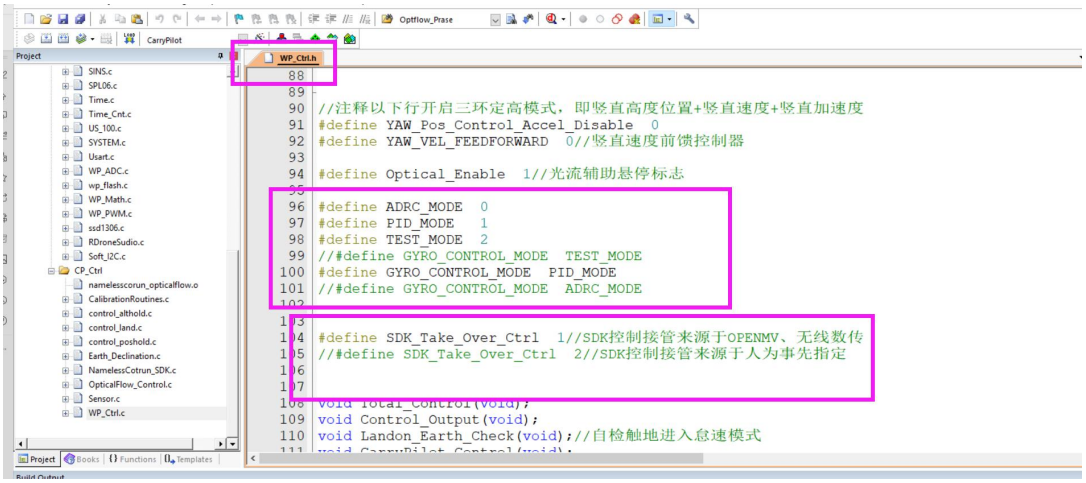
飞控学习来讲，需要结合实际不同的机架类型、动力配置来对飞控参数进行整定，来实现稳定飞行，因此调参是必会的一门学问。如果个人对于飞行手感有要求的，就要调的一手好参数。无名飞控默认的参数是样机 F330 的参数，实际默认样机参数可以让 F380、F450、F650 动力装基本飞行，对于其它配置的飞机，需要自行对控制参数进行优化。飞控可以利用匿名上位机对飞行器参数进行调整，实际调试界面如下。

首次使用时，打开飞控设置子窗口，点击读取 PID，即可获取飞控当前的 PID 参数，

用户可以在对应框输入调整值，点击写入 PID 后，当前 PID 参数会刷新并将刷新后的 PID



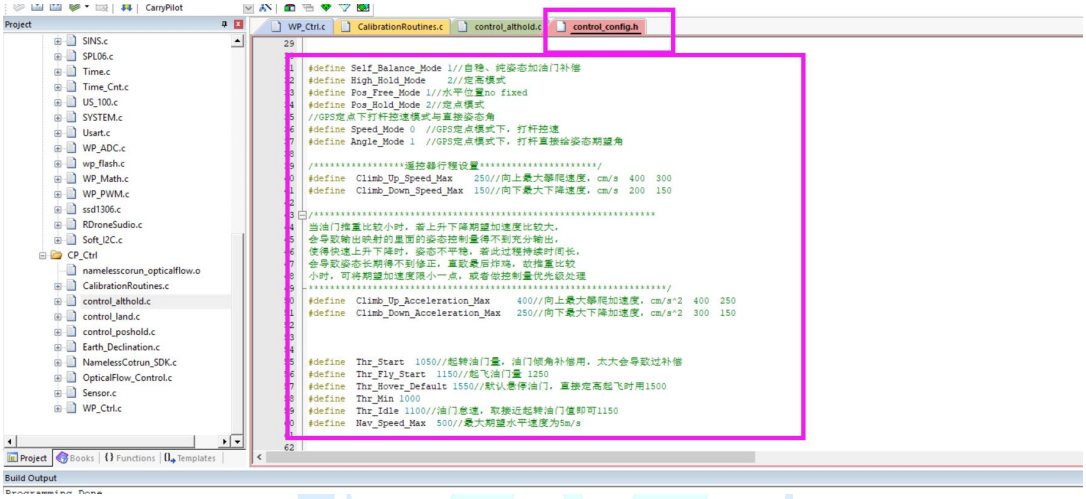
参数保存与 Flash 中（只要下载时不点击擦除全部芯片，下次上电时数据仍在）。如果客户在调试过程中想要恢复初始默认参数点击恢复默认 PID 即可。注意实际控制 PID 参数为地面站所显示的 PID 参数的 1/100，位置控制的位置、速度参数为经、纬度方向共用，PID11 为定高控制加速度环参数，实际设计控制器是不仅仅是设置 PID 参数，也需要对偏差、积分、总输出等细节部分进行处理等。更多样机参数见售后群文件。



十一、常见参数更改

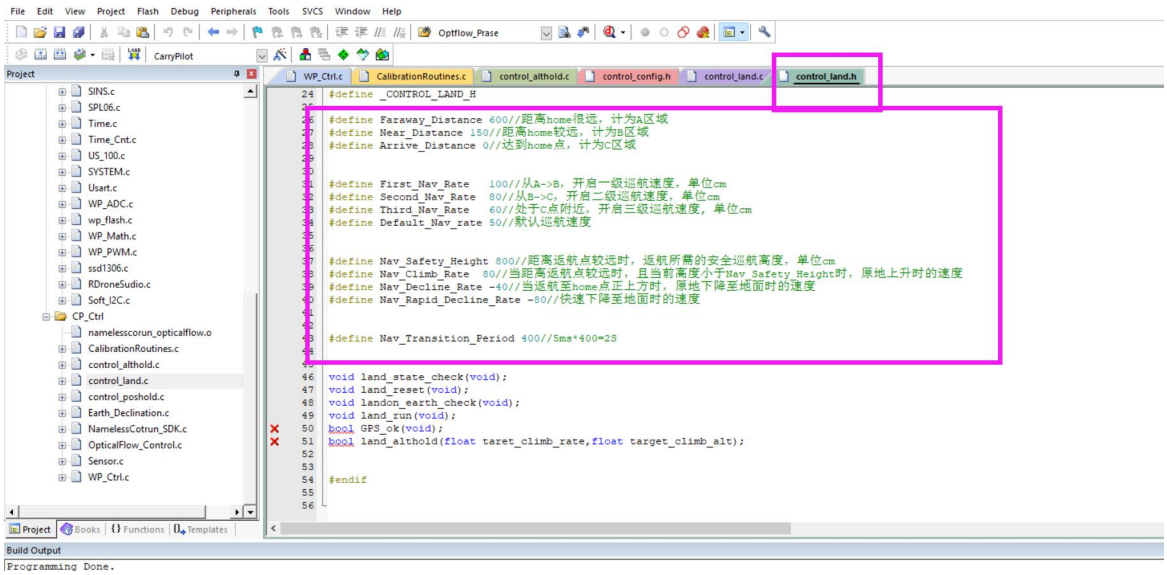
1. control_config.h

- 更改最大上升、下降的速度、加速度
- 定点模式下，打杆控速度/控姿态
- 更改电机起转油门、起飞油门、油门最小值、电机怠速值、最大水平巡航速度，一键起飞的目标高度等。



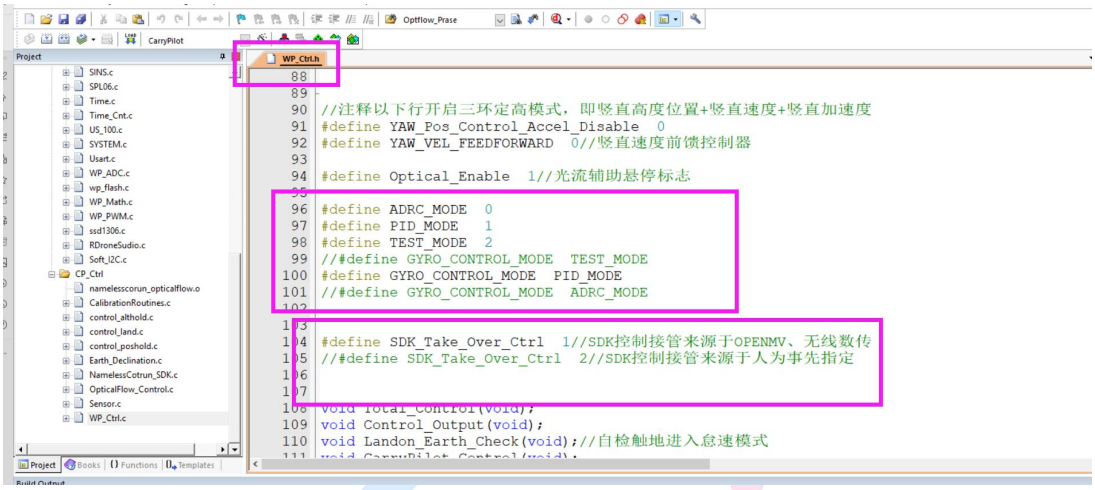
2. control_Land.h

- 更改返航安全高度、返航上升速度、返航巡航速度、区域半径、过度缓冲时间等



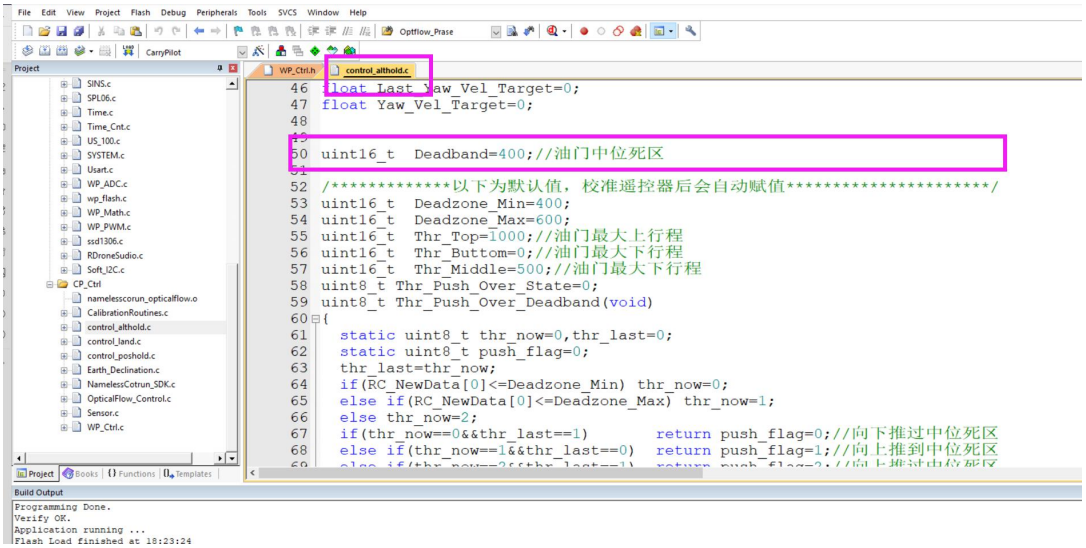
3. NamelessCotrun_Ctrl.h

- 为了方便新手学习，这里可以更改是否加入速度前馈控制、是否采用三（两）环定高控制，默认为前者。
- 姿态角速度控制器采用 PID 或者 ADRC。



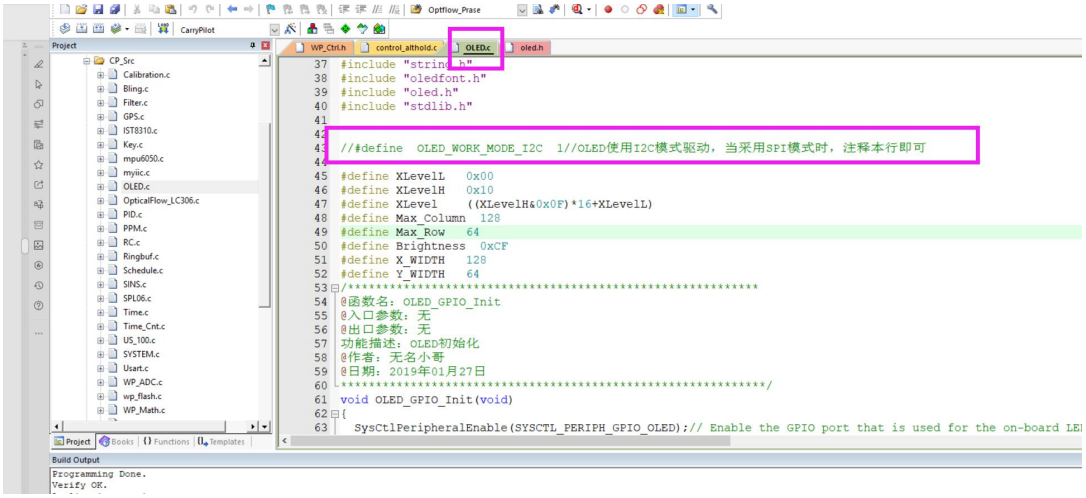
4. control_althold.c

- 为了方便新手操作，可以把油门中位死区改大一点，默认为 400



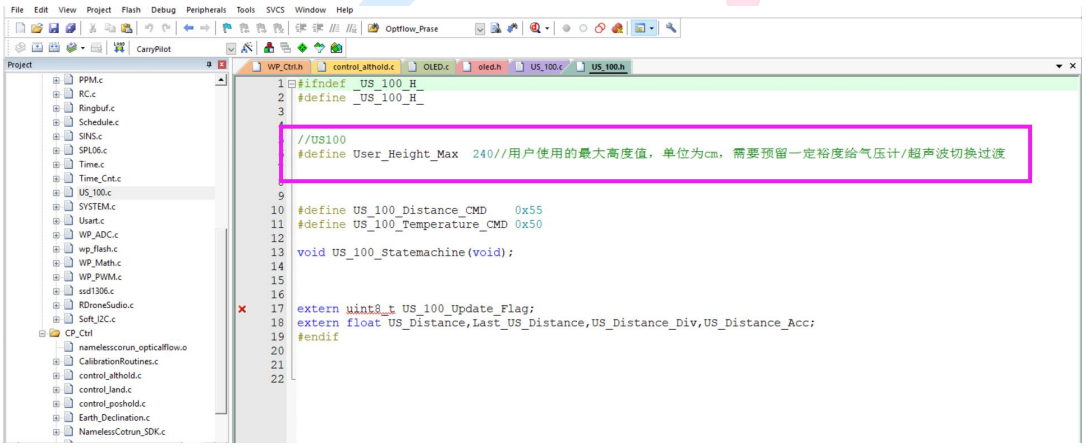
5. OLED.c

- 4 线 I2C、7 线 SPI 驱动 OLED 屏幕切换，根据注释改下宏定义即可



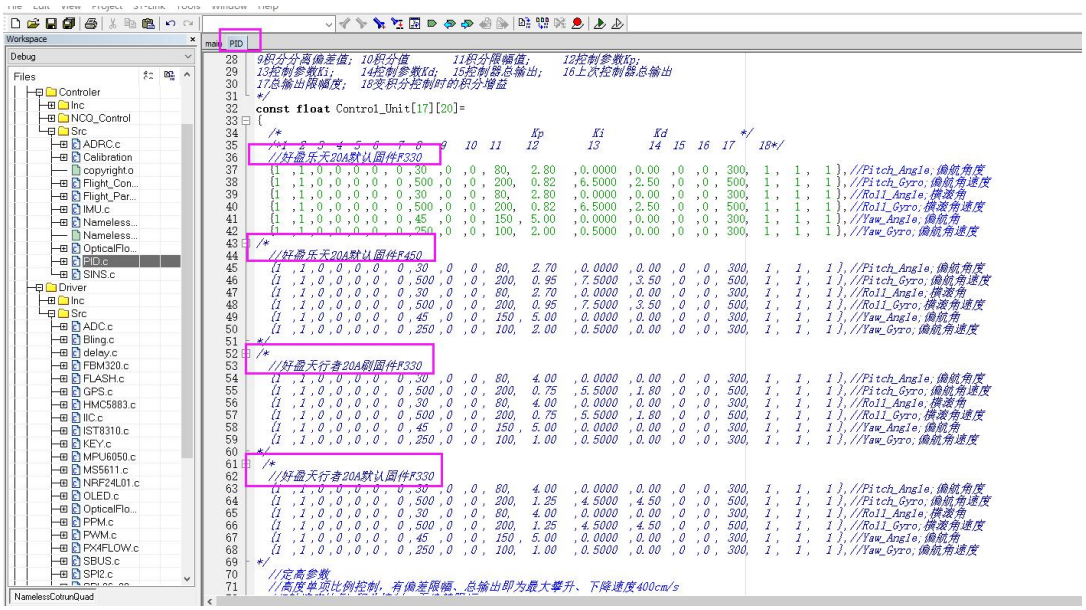
6. US_100.h

● 根据实际使用的超声波模块，强烈推荐 US100，设置最大有效高度与最大使用高度，根据注释改下宏定义即可



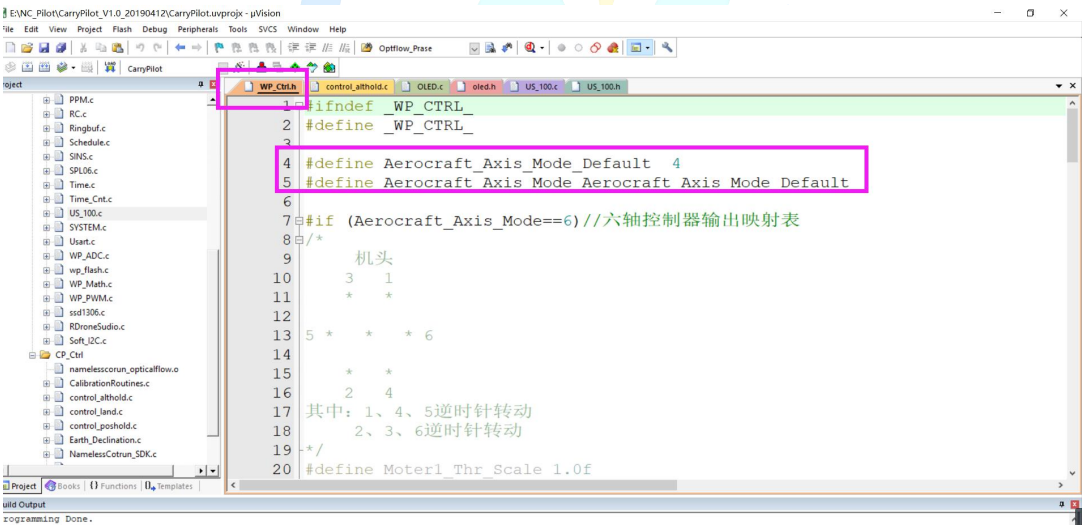
7. PID.c

● 根据实际动力装配置，结合不同轴距、电调选取默认样机参数，根据注释打开/关闭对应注释即可，更多动力装参数见售后群客户分享。

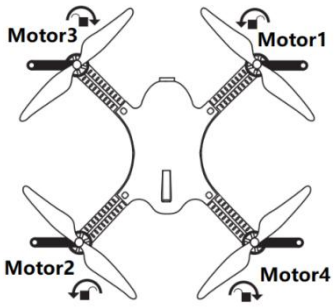


8. NamelessCorun_Ctrl.h

- 飞控作为六轴使用时，将下图宏定义出改为
- #define Aerocraft_Axis_Mode_Default 6 即可



十二、飞控操作使用入门



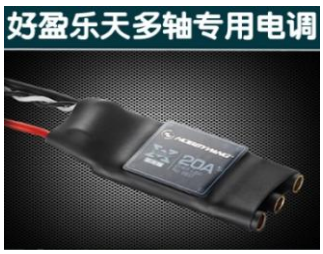
1、竖直箭头为机头方向，安装时候需要与传感器盖子上箭头方向一致

2、图中 1#与 2#电机逆时针转动，3#与 4#电机顺时针转动，如果安装后上电发现电机旋转方向与图中不一致，任意调换电机、电调连接线的的其中两根即可。桨的方向需要和上图中一致，电调信号输入务必按照飞控板上所示 1、2、3、4 连接。

3、本程序默认参数为样机配置的参数，不同动力装需要对控制参数进行整定，样机动力装如下：

机架：F330 沉金板

电调：天行者电调 20A（需刷固件，偶尔会掉固件，重刷即可）、不愿意刷固件者，推荐使用更适合多旋翼的高性价比**好盈乐天 20A 电调**，土豪人民币玩家直接选铂金 30A。



电机：2212 KV930 电机

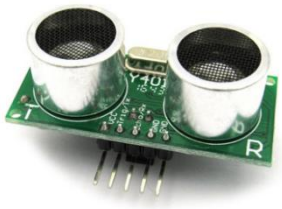
螺旋桨：乾丰 8045 螺旋桨

锂电池：红牌 25C 2200mah 3S

航模遥控器：FS I6（自刷 10 通道）+接收机（IA6B/FS-RX2A Pro）

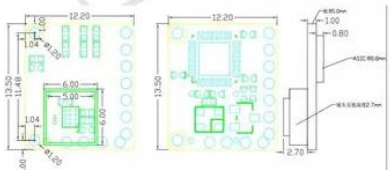
GPS 模块：M8N 模块

超声波模块：HC_SR04、US100 模块（推荐）



光流模块：LC306 光流定位模块（湖南优象——零度智控持股）

图 1 UP-FLOW-LC-306 产品结构图



3DR 数传模块

飞控装机完毕后（装机方法见如下教程、视频）：

史上最详细的 F450 四轴装机实例：

http://www.modouwo.com/ADiscuss/ListDetail_3_51_3113.html

多轴飞行器 DIY 教程：

http://www.modouwo.com/ADocumentMenu/DocumentMenu_DocunemtMenuStudy.html

四轴多旋翼无人机组装：

https://v.youku.com/v_show/id_XMTQ5MTA0NTE4MA==.html?spm=a2h0k.11417342.soresults.dtitle

极飞科技 CEO 彭斌教你组无人机：

http://vo.youku.com/v_show/id_XODcxODA1MDQ=.html?spm=a2h0k.11417342.soresults.dtitle

十三、极飞发展历程

2007 极飞科技（XAIRCRAFT）成立，开始无人机飞行控制系统的研发

2010 第一代里程碑产品 X650 四轴飞行器问世

2012 发布 SUPERX 飞行控制系统，开始无人机行业应用的探索

2013 开始探索无人机在农业领域的应用，在新疆开始植保无人机喷洒实验 2014 极飞农业（XPLANET）成立，在新疆完成万亩棉花地的植保服务运营 2015 发布第一代 P20 植保无人机系统，聚焦农业无人机研发

2016 发布 P20 2017 款植保无人机系统和 SUPERX2 RTK 飞行控制系统 成立极飞地理（XGEOMATICS）和极飞学院（XAIRCRAFT ACADEMY）

2017 发布 3 款全新植保无人机系统 （P10 2018 款、P20 2018 款、P30 2018 款） 发布极飞地理智能测绘无人机 C2000 发布极飞物联智能农田监测站 FM1 发布无人机共享租赁平台、农事服务平台和农业 AI 引擎（XAI）成立极飞日本株式会社，极飞学院落地澳洲。

通过研究无人机、物联网、云计算和人工智能技术，不断为人类简化生产方式让农业变得更智能、更高效、更环保。

我们的理想，是要让全人类的农业生产，从粗放、盲目、只追求效率和产值的方式，向智能、精准、可持续发展的方式转型。

赋予无人机更高的科技使命，而不只是一个农业生产工具。我们正在把“种地”变成一件很酷的事情。

做智慧农业，是一件非常有社会价值的事情，它可以减少污染、改善生产，保障我们的食品安全。

大疆创新 CEO 汪韬教你玩转 F330：

https://v.youku.com/v_show/id_XMzE5MzQ3Nzg0.html?spm=a2hzp.8253869.0.0

性能测试合集：

https://v.youku.com/v_show/id_XMzEyNzczNDUy.html?spm=a2hzp.8253869.0.0

十四、大疆发展历程

2006 破茧而出，DJI 于 2006 年 11 月 6 日扬帆起航 正式踏上征程。

2007 推出直升机飞控 XP2.0 大疆飞控第一次达成超视距飞行。

2008 大疆开发第一架自动化电动无人直升机 EH-1 直升机飞控 XP3.1 第三代产品研发成功 奔赴汶川地震灾区，采集第一手图像资料 大疆搬进西丽创业园的办公大楼。

2009 珠峰高原测试成功 展翅西藏，突破技术疆界。

2010 XP3.1 地面站诞生 重磅产品 ACE ONE 诞生 销售额猛增，享誉国际。

2011 ACE WAYPOINT 新一代地面站问世 WOOKONG-H 直升机飞控破石而出 WOOKONG-M 多旋翼飞控腾空出世 风火轮正式发布，大疆迈入消费级模型市场 多旋翼王子 “NAZA-M” 诞生。

2012 世界首款航拍一体机---飞翔之精灵 PHANTOM 出世 世界首款专业一体化多旋翼飞行器筋斗云 S800 诞生 风火轮系列 F330、F550 发布，拓宽消费级模型市场 专业三轴云台禅思 ZENMUSE 震撼发布 重量级直升机飞控产品 ZA-H 诞生 IOSD&5.8G 图传上市

美国及德国分公司成立。

2013 推出全球首款会飞的照相机 DJI 明星产品精灵 PHANTOM 2 VISION 引领全球航拍热潮。

2014 誉满世界明星产品精灵 PHANTOM 2 VISION + 诞生 推出世界首款自带 4K 相机的可变形航拍器 “悟” INSPIRE 1 推出全高清数字图像传输系统 LIGHTBRIDGE 筋斗云 S1000 全方位改进升级，性能更加卓越 为专业摄影师定制开发三轴手持云台系统 “如影” RONIN SDK 软件开发套件助力开发者扩展航拍应用领域。

2015 精灵 PHANTOM 3 系列问世，推动全球消费级无人机市场变革 一体化手持云台相机灵眸 OSMO 面世 发布 MG-1 农业植保机，正式步入农业无人机领域 “如影” RONIN-M 赋予全世界摄影师更多创造空间 ONBOARD SDK、开发者平台经纬 M100 与视觉传感导航系统 GUIDANCE 为无人机软硬件应用创新打开广阔空间 推出全球首款 M4/3 航拍相机禅思 X5 系列 禅思 XT 突破光线与空间的局限，随时随地地捕捉清晰、精准的热成像。

2016 重磅推出精灵 PHANTOM 4，开启机器视觉的时代。
推出 A3 飞控，开创更多可能，全面满足行业应用需求 顶级飞行平台经纬 M600 为影视航拍与行业应用带来新突破 衔接天地，“如影”RONIN-MX 轻松应对各类拍摄需求 灵眸 OSMO+和手机云台将稳定影像带给每一个用户 创造随身无人机“御”MAVIC PRO,将先进技术浓缩至小巧机身 精灵 PHANTOM 4 PRO 将精灵系列提升至专业级设备行列 “悟”INSPIRE 2 为影视制作提供高画质、高性能的无人机。

无名小哥寄语：让热爱飞行的人们紧密相连，坚持梦想，深耕细作，砥砺前行、直至拨云见日，一往无前，为无人机前辈们对行业的真爱点赞！！！

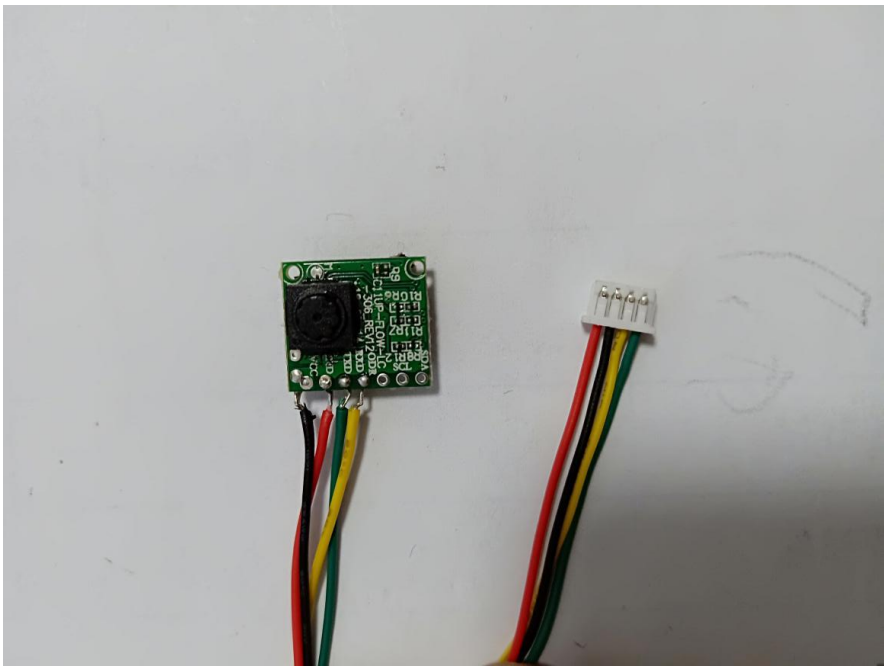
十五、标准样机装机连线图

整机到手飞客户请按上图装好正反桨，务必拧紧，防止射桨炸机

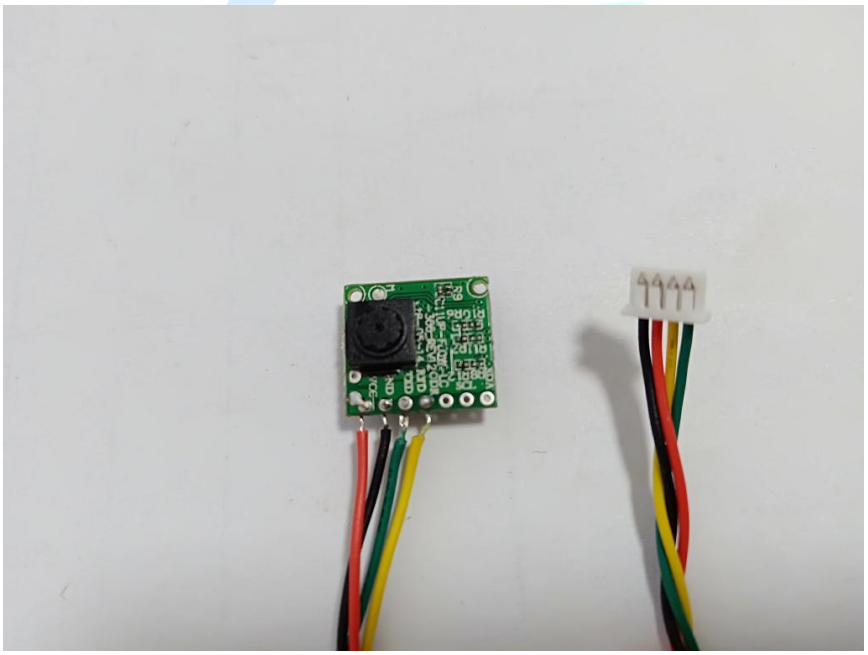
十六、光流模块安装细节与方向

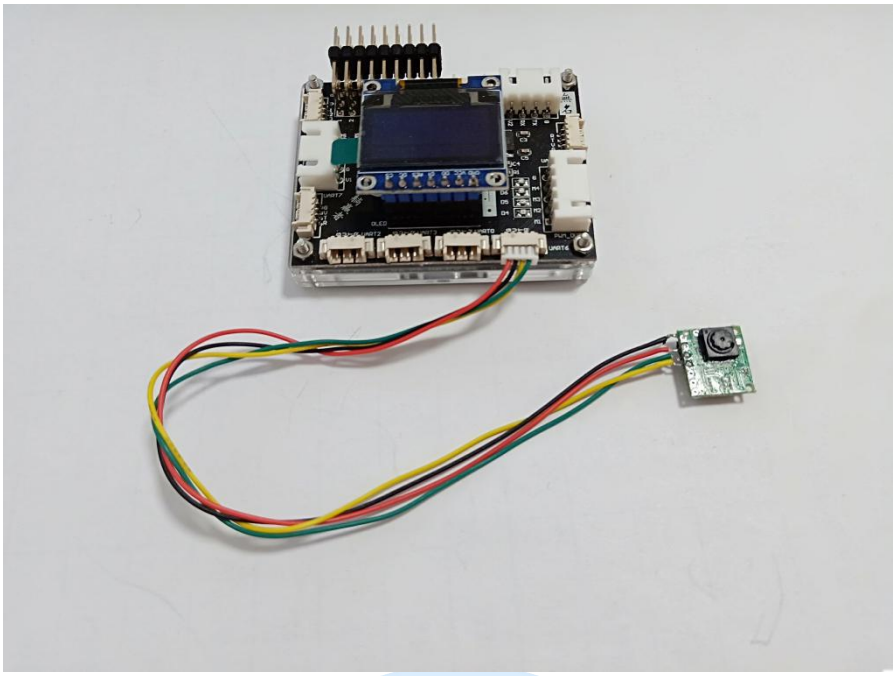
① 光流传感器属于明感原件，无过压、反接保护，在使用前请务必详细阅读使用说明书，注意电源顺序。

- 非整机客户单独购买光流自行组装时，最简焊接（推荐）方式如下（无需调换插口端 1.25mm 4P 线序）：

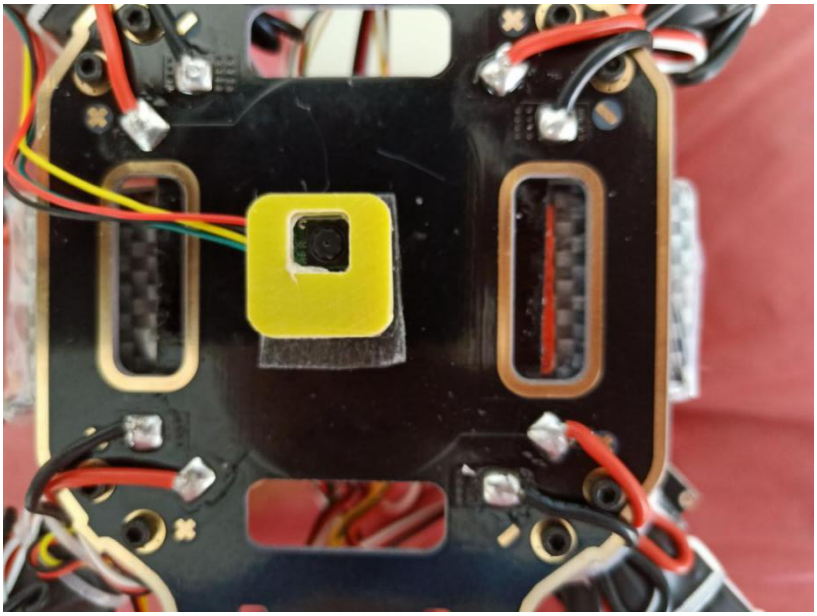


- “强迫症患者”会严格把电源顺序红色作为+，黑色最为-，可以自行调换插口端黑红线序，**务必核对-核对-再核对后**，再上电。光流模块接串口 6。

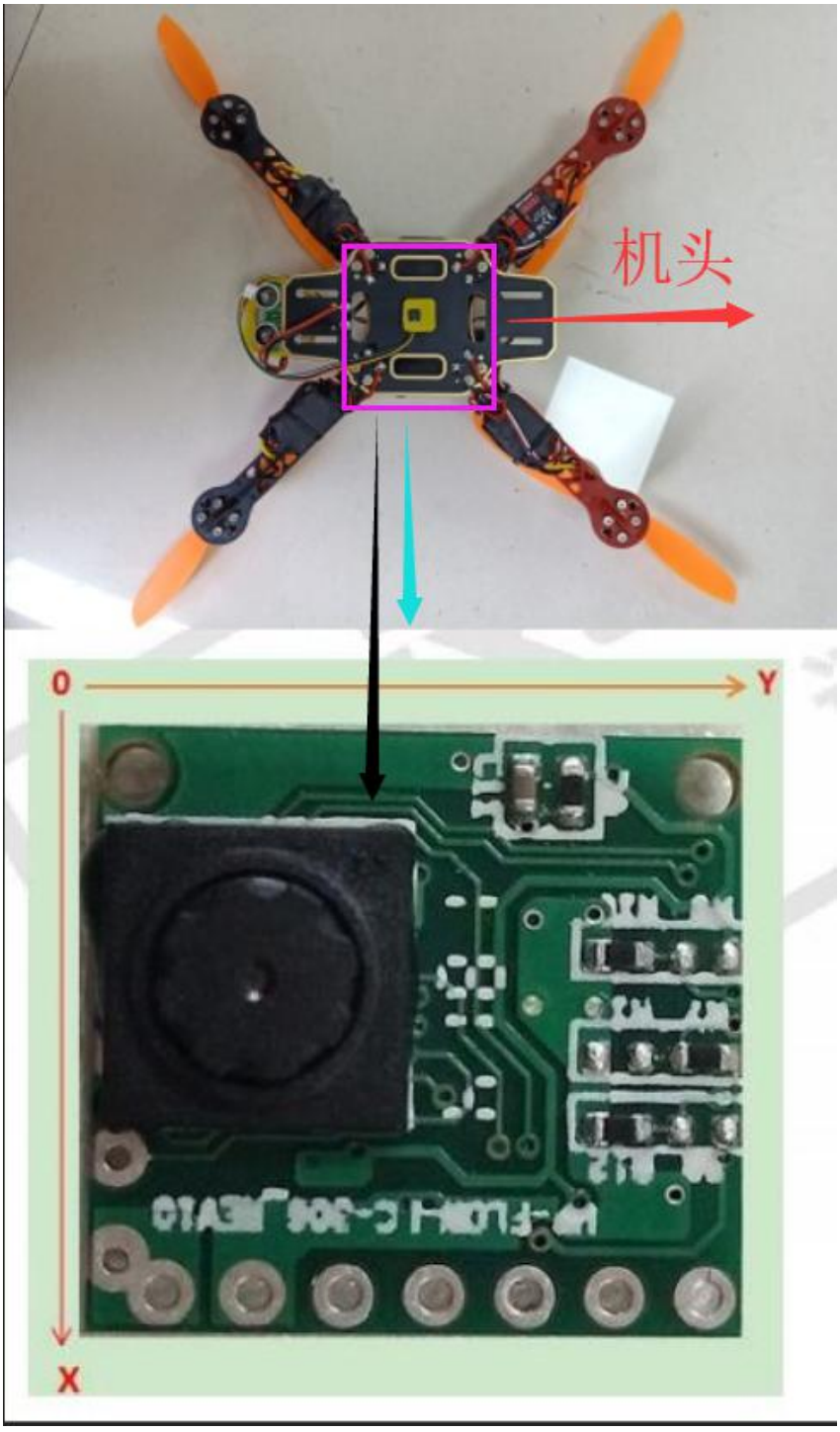


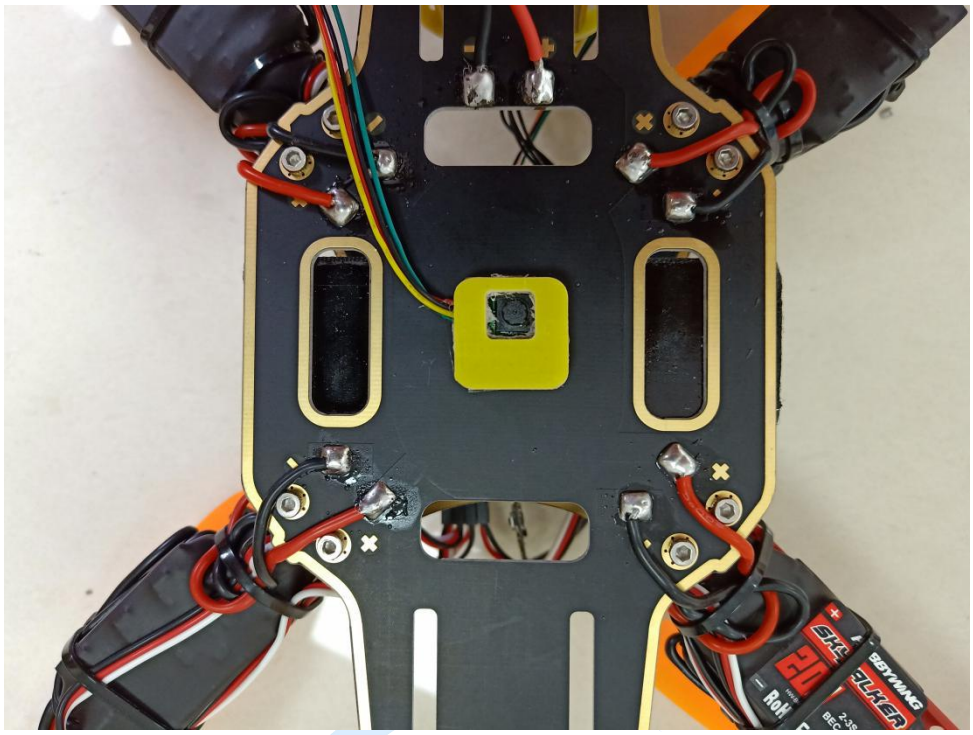


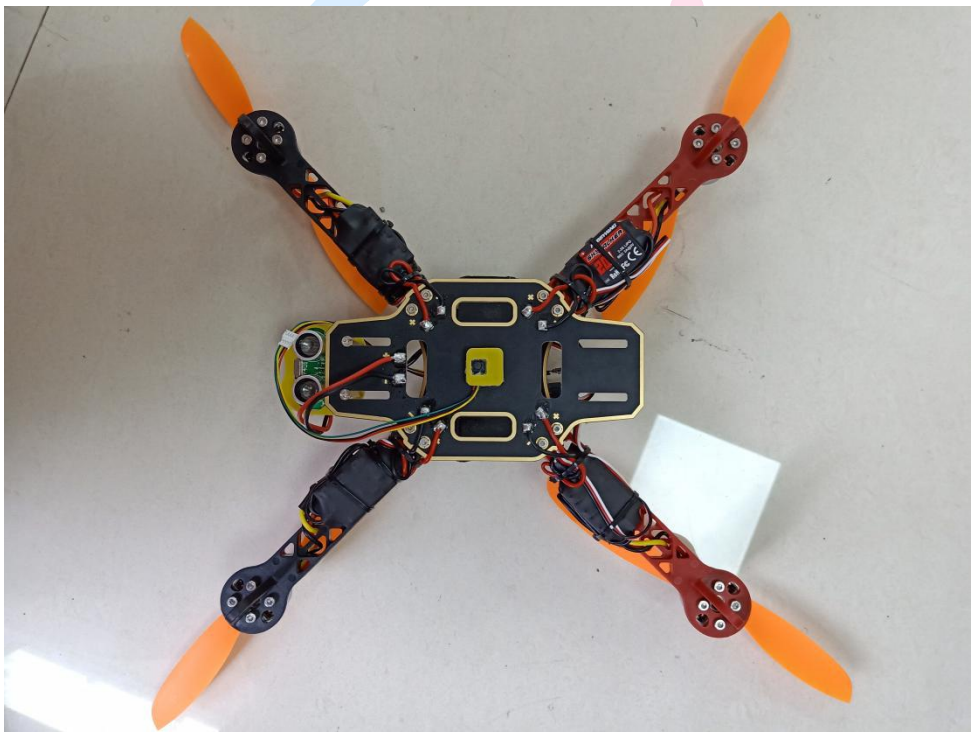
② 光流模块尽量安装在机体中心，背面贴 3M 胶示意图如下：



③ 光流模块与无名飞控方位图如下图


机头
O
Y
X

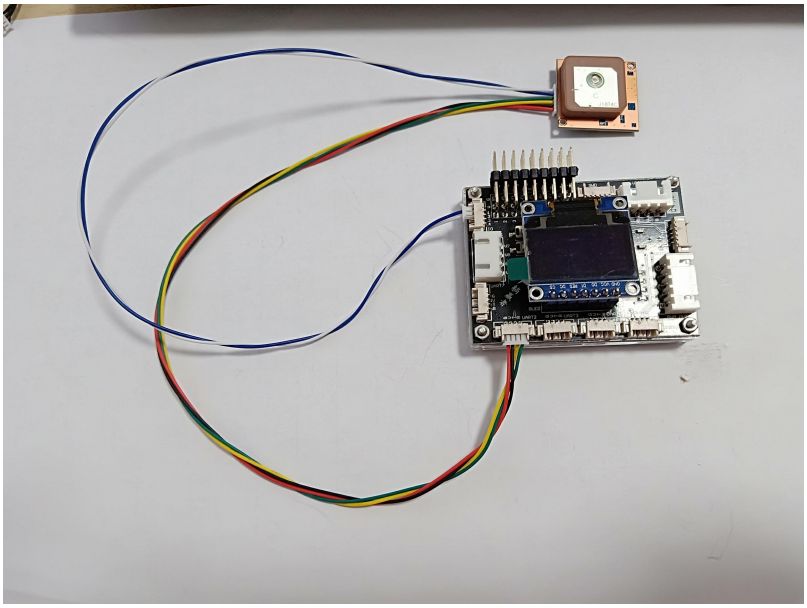




十七、GPS 连接与安装

飞控支持户外 GPS 定点、定速巡航、返航等功能，外接 GPS 模块接飞控 UART0 端口，飞控支持 M8N GPS 模块，对于无外部磁力计的 GPS 而言，GPS 安装无朝向可言，只需要保证 GPS 陶瓷天线朝上即可。

当需要用到外部磁力计时，磁力计数据可以通过预留 IO 口接入飞控，磁力计原始数据需要根据和飞控机头相对角度，进行重映射。结合磁力计类型、安装方向自行调整。



十八、首飞前的飞控校准与基本飞行方法

1、需要按如下操作步骤要求依次完成如下操作（必须按照此顺序严格执行）

① 第一步校准遥控器行程，按照前文所述方法操作即可。（必须完成遥控器行程校准，因为后续操作都依赖于校准的遥控器行程，校准前请先确定通道顺序、是否需要反向操作），务必搞清楚自己的遥控器 5、6、7、8 通道对于遥控器上的哪一个开关/旋钮，默认位置是处于高位、还是低位，5、6、7、8 通道对应着飞控的模式设置，不清楚 5、6、7、8 通道特性无法正常使用飞控。

https://www.bilibili.com/video/av38892290

② 第二步校准电调行程，注意安全，一定要卸桨—卸桨—卸桨再操作（新手不要

心血来潮，一腔热血地装好飞机，啥都不管，不按要求步骤来就去做测试，当心血肉模糊、血肉横飞、血流成河，最后眼中流血，心里成灰），按照前文所述方法操作即可。

https://www.bilibili.com/video/av38892453

③ 第三步校准加速度计，六面放置时尽可能保证该面水平静置，按照前文所述方法操作即可。

https://www.bilibili.com/video/av38892352

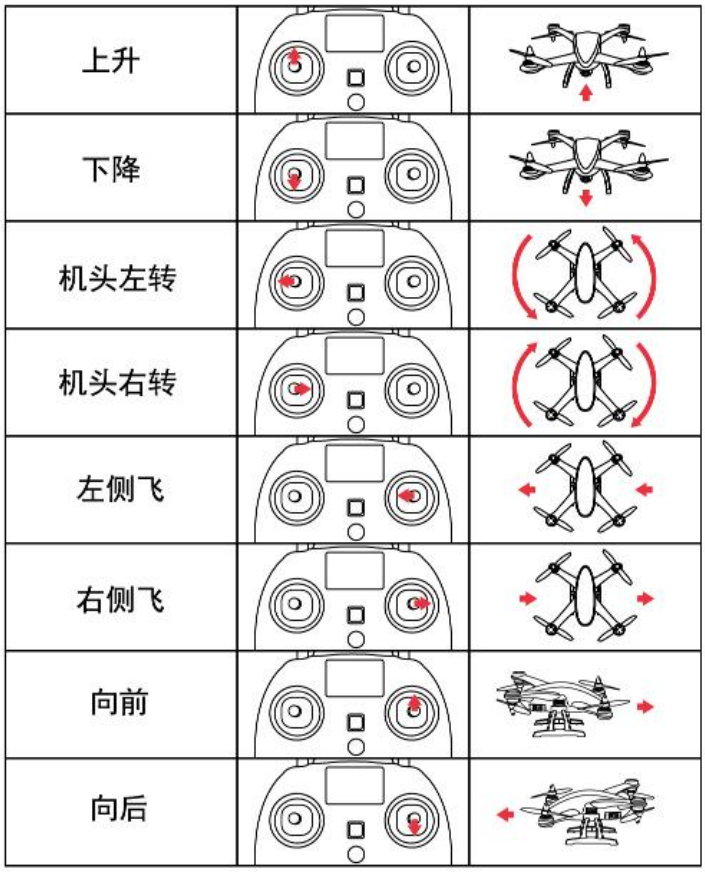
④ 第四步校准磁力计，去户外磁场干扰小的地方（远离电梯等强磁设备）均匀转动无人机，按照前文所述方法操作即可。

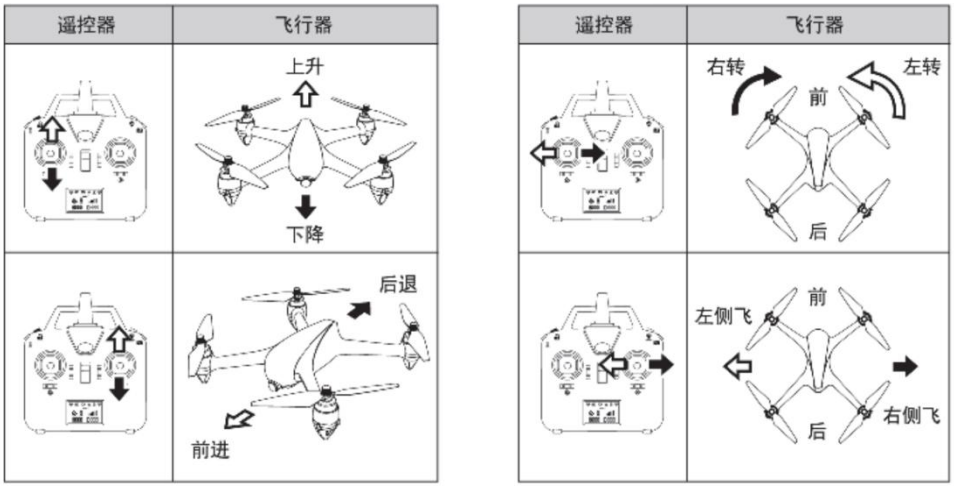
https://www.bilibili.com/video/av38892393

⑤ 第五步校准机架水平，确保无人机放置于水平地面后再执行此操作，按照前文所述方法操作即可。

⑥ 默认动力装用默认控制参数即可实现爽飞，不同动力装需要自行优化参数，售后群有购买客户提供的不同动力装爽飞方案与参数，可结合自己要求自行配置，新手强烈推荐直接使用默认动力装，避免初始学习挫败感太强而弃模。

2、用户需起飞前，需掌握基本飞行操作，基本操作如下图。





3、飞控默认模式

本程序第 5 通道为定高通道、第 6 通道为 GPS 定点通道、第 7 通道为 GPS 模式下一键返航、非 GPS 模式下一件降落通道。当遥控器第 5、6 通道为在低位时默认纯姿态自稳模式，飞行起飞时为自稳模式，确保自稳状态下飞机飞行稳定后，飞机起飞到空中，找准基本悬停油门后（油门升力与重力接近的油门杆位，此悬停油门测试出来后可在程序里面直接写死，这样就可以直接定高起飞，TIVA 飞控支持直接定高起飞，悬停油门默认 1500，此值可以结合实际动力装自行修改，新手在操作定高起飞前，需确保单姿态控制的参数已经调试 OKAY），拨动第 5 通道至高位,此时飞控进入定高模式，油门杆回中表示保持当前高度，向上推过中位死区后，表示给定向上运动速度，同理，向下推过中位死区后，表示给定向下速度，油门杆回中即可保持当前高度位置。

飞机进入定高模式后，拨动第 6 通道至高位，飞机进入 GPS 定点模式，确保拨动定点模式是在户外（室内禁用、远离磁场干扰区、GPS 近地面信号质量会比较差），在 GPS 定点模式下，方向杆打杆有两种模式：1、遥感直接对应姿态期望角度；2、遥感对应给定无人机机体方向上的运动期望速度（程序默认为此模式）。两种模式直接更改程序相关宏定义即可，也可以采用遥控器某个通道使得这两种模式来回切换。（GPS 设置输出频率 10Hz，波特率 921600，只解析 PVT 语句，相关设置见教学视频）

飞控第 7 通道为一键返航着陆通道，1、在 GPS 满足定位条件下，无人机会执行返航操作，当距离 home 点较远时，若当前高度不满足安全返航高度（默认 8 米）时，无人机会首先原定攀升到安全高度，若距离 home 点比较近时，无人机会保持当前高度，直接飞到 Home 点正上方后，再下降至地面。执行返航操作。返航过程中不需要人为遥控操作无人机，无人机会自动做出决策去控制飞行。同时当存在手动操作时（比如前方

有障碍、着陆时因 GPS 漂移，home 点的地面不平等），需人为手动纠偏时，无人机会进入 GPS 定点模式下的手动控制模式，待人为操作消失后，无人机会再次判断当前状态，继续执行返航操作。2、在 GPS 满足定位条件下，无人机会直接原地下降，同理，再自动下降过程中，仍然可以手动操作无人机。

遥控器第 8 通道为光流定点悬停模式操作通道（高位有效），光流悬停模式只有再定高模式下（第 5 通道处于高位、第 6 通道处于低位）有效，即纯姿态自稳模式与 GPS 定点模式下，切第 8 通道将不会进入光流悬停模式。

无人机一键定高起飞操作，无人机开机置于地面，将第 5 通道切到高位进入定高模式，对飞控进行解锁操作后，然后针对非回中式油门，需要人为将遥控器置与推到中位死区内（自然回中式油门本操作可省略），此时无人机在地面处于怠速模式，短时间内，在高位与低位之间来回往复切换遥控器第 8 通道 3 次，无人机即可完成一键起飞到目标高度自动悬停。其中起飞的目标高度是以切换瞬间相对地面高度值，默认为 1m，可根据不同应用要求自行修改。注意当存在超声波定高时，一键起飞高度不要超过超声波量程。起飞上升过程中可调节水平位置、若起飞过程中存在油门上下回中操作，自动一键起飞过程立刻结束，进入正常定高控制模式。（一键起飞操作是建立在定高功能正常的情况下，需先确定手动定高起飞正常后，再执行一键起飞操作，若对此操作过程有不详尽的地方见教学视频）

新手必看的直接定高起飞教程：

https://www.bilibili.com/video/av31341679

新手必看的一键定高起飞、降落教程：

https://www.bilibili.com/video/av29889695

十七、注意事项：

1、飞行器时，请确保飞行器重心在机架中心，有负载的在机架中心的垂直方向上。

2、 安装主控器时，尽量安装在靠近中心位置，确保主控印有标记的一面朝上，并使其与机身水平面保持平行，否则会导致飞行

器水平方向飘移。

3、 主控器安装有方向要求，务必使箭头的朝向与飞行器机头方向一致。

4、 在固件升级、调试过程中请断开电调与电池的连接或移除所有桨翼！

5、 飞行时切记先打开遥控器，然后启动多旋翼飞行器！着陆后先关闭飞行器，再关闭遥控器！

6、 切勿将油门的失控保护位置设置在 10%满量程以下。

7、 在飞行过程中油门杆量须始终距熄火位置 10%满量程以上！

8、 低压保护的目的不是娱乐！在任何一种保护情况下，您都应该尽快降落飞行器，以避免坠机等严重后果！

9、 GPS 与指南针模块为磁性敏感设备，应远离所有其他电子设备。

10、 GPS 模块为选配模块(非标配)，请选用此模块的用户关注说明书中涉及 GPS 的内容,未选用此模块的用户请忽略 GPS 控制模式下的相关内容。

11、 强烈建议将接收机安装到机身板下面，天线朝下且无遮挡，以避免无线信号因遮挡丢失，而造成失控。

12、飞行前请检查所有连线正确，并且确保连线接触良好。

13、使用无线视频设备时，安装位置请尽量远离主控系统（>25cm），以避免天线对主控器造成干扰。

十八、扩展功能

18.1 OpenMV 前言

OpenMV 是基于 MicroPython 的嵌入式机器视觉模块，目标是成为机器视觉界的“Arduino”。它成本低易拓展，开发环境友好，除了用于图像处理外，还可以用 Python 调用其硬件资源，进行 I/O 控制，与现实世界进行交互。

你想学习机器视觉,却不知从何入手？在傅立叶变换，小波变换等一系列信号与系统的强烈攻势之下，挣扎抑或挫败？

封装各种算法细节，不需要直接跟底层代码打交道。

用户友好的 PythonAPI，让你开始的时候，就享受机器视觉带给你的愉悦，逐层剥开底层算法实现原理，代码开源，随意修改底层源码，编译固件。

非计算机专业，想在自己的机器人或者小车上加载机器视觉模块，却担心自己学不懂？电赛来袭，慌忙准备，没时间系统学习机器视觉？

来用 OpenMV 吧，快速入手，不需要专业的背景知识。

几行代码，轻松搞定。例程丰富，也许你需要做的只是改一下参数，让你的机器人开启视觉智能。

18.2 OpenMV 简介及开发语言介绍

18.2.1 OpenMV 简介

Openmv 是使用 STM32F765VI ARMCortex M7 处理器，216MHz，512KBRAM，2Mbflash，所有的 I/O 引脚输出 3.3V 并且 5V 兼容。这个处理器有以下的 IO 接口：

全速 USB(12Mbs)接口，连接到电脑。当插入 OpenMV 摄像头后，你的电脑会出现一个虚拟 COM 端口和一个“U 盘”。SD 卡槽拥有 100Mbs 读写，这允许你的 OpenMV 摄像头录制视频，和把机器视觉的素材从 SD 卡提取出来。一个 SPI 总线高达 54Mbs 速度，允许你简单的把图像流数据传给 LCD 扩展板，WiFi 扩展板，或者其他控制器。一个 I2C 总线，CAN 总线,和一个异步串口总线(TX/RX)，用来链接其他控制器或者传感器。一个 12-bitADC 和一个 12-bitDAC。3 个 I/O 引脚用于舵机控制。所有的 IO 口都可以用于中断和 PWM(板子上有 10 个 I/O 引脚)。一个 RGBLED（三色），两个高亮的 850nm IR LED（红外）.

OV7725 感光元件在 80FPS 下可以处理 640x480 分辨率 8-bit 灰度图或者 320x240 分

辨率 16-bit RGB565 彩色图像。当分辨率低于 320×240 可以达到 120FPS。大多数简单的算法运行在 30FPS 一下。

OpenMV1 的像素是 30 万，OpenMV2 的像素是 200 万，OpenMV3 又改为了 30 万像素。即便使用了 STM32 最好的芯片(STM32F7)，在处理图像的时候，面对大量的计算，也难免会有些吃力。OpenMV3 之所以使用低像素，是因为在清晰度和性能之间做了一个折中。30 万像素，在处理一些图像细节的时候会比较吃力，但是巡线，颜色追踪还是没有问题的。

18.2.2 应用

目前 OpenMV 摄像头可以用来做一下的事情(未来会更多)：

Frame Differencing 帧差分算法。

你可以使用 OpenMV Cam 上的帧差分算法来查看场景中的运动情况。帧差分算法可以将 OpenMV 用于安全应用。

Color Tracking 颜色追踪

你可以使用 OpenMV 在图像中一次检测多达 16 种颜色（实际上永远不会想要找到超过 4 种颜色），并且每种颜色都可以有任意数量的不同的斑点。OpenMV 会告诉您每个 Blob 的位置，大小，中心和方向。使用颜色跟踪，您的 OpenMV Cam 可以进行编程，以跟踪太阳，线跟踪，目标跟踪等等（视频演示详见：https://singtown.com/video/。）。

Marker Tracking 标记跟踪

您可以使用 OpenMV Cam 来检测颜色组的颜色，而不是单独的颜色。这允许你在对象上放置颜色标签（2 种或多种颜色的标签），OpenMV 会获取标签对象的内容（视频演示详见：https://singtown.com/video/。）。

Face Detection 人脸检测

你可以使用 OpenMV Cam（或任何通用对象）检测脸。OpenMV 摄像头可以处理 Haar 模板进行通用对象检测，并配有内置的 Frontal Face 模板和 Eye Haar 模板来检测人脸和眼睛 Eye Tracking 眼动跟踪。你可以使用眼动跟踪来检测某人的注视方向。你可以使用它来控制机器人。眼睛跟踪检、测瞳孔的位置，同时检测图像中是否有眼睛。

Optical Flow 光流

你可以使用光流来检测您的 OpenMV 摄像机面前的画面。例如，您可以使用四旋翼

上的光流定点来保证在空中的稳定性。

QR Code Detection/Decoding 二维码检测/解码

您可以使用 OpenMV Cam 在其视野中读取 QR 码。通过 QR 码检测/解码，您可以使智能机器人能够读取环境中的标签。请关注我们后期的代码更新

Data Matrix Detection/Decoding 矩阵码检测/解码

OpenMV Cam M7 也可以检测和解码矩阵码（2D 条形码）。请关注我们后期的代码更新。

Linear Barcode Decoding 条形码

OpenMV Cam M7 还可以处理 1D 条形码。他可以解码 EAN2、EAN5、EAN8、UPCE、ISBN10、UPCA、EAN13、ISBN13、I25、DATABA、DARABAR_EXP、CODABAR, CODE39、CODE93 和 CODE128（视频演示详见：https://singtown.com/video/ ）。

AprilTag Tracking 标记跟踪

甚至比上面的 QR 码更好，OpenMV Cam M7 也可以追踪 160x120 的 AprilTags，高达约 12FPS。AprilTags 是旋转不变，尺度不变，剪切不变和照明不变的最先进的基准标记（视频演示详见：https://singtown.com/video/。）。

Line Detection 直线检测

OpenMV Cam 可以在几乎跑满帧率的情况下，快速完成无限长的直线检测。而且，也可以找到非无限长的线段或者使用霍夫变换检测间断的线（视频演示详见：https://singtown.com/video/。）。

Template Matching 模板匹配

您可以使用 OpenMV 模板匹配来检测视野中是否有模板相似的图片。例如，可以使用模板匹配来查找 PCB 上的标记，或读取显示器上的已知数字。

Image Capture 图像捕捉

你可以使用 OpenMV 捕获高达 320x240 RGB565（或 640x480 灰度）BMP/JPG/PPM/PGM 图像。可以直接在 Python 脚本中控制如何捕获图像。最重要的是，使用机器视觉的算法，进行绘制直线，绘制字符，然后保存。

Video Recording 视频录制

您可以使用 OpenMV 摄像机记录多达 320x240 RGB565（或 640x480 灰度 MJPEG 视频或 GIF 图像。您可以在 Python 脚本中直接控制如何将每个视频帧记录，并完全控制视

频录制的开始和结束。而且，像拍摄图像一样，您可以使用机器视觉的算法，进行绘制直线，绘制字符，然后保存。最后，所有上述功能都可以混合 IO 引脚的控制，来配合你自己的自定义应用，以与现实世界交谈。

18.2.3 Python 开发语言入门

廖学峰-小白的 Python 新手教程：https://www.liaoxuefeng.com/wiki/0014316089557264a6b348958f449949df42a6d3a2e542c000。

廖学峰大大的 Python 教程，我见过的写得最好的 Python 开发教程,个人的 Python 入门就是从廖学峰的教程开始的。笨办法学 Python：https://wizardforcel.gitbooks.io/lpthw/content/。此书有大量编程任务,在 Python 中通过练习和记忆等技巧慢慢建设和建立技能，然后应用它们解决越来越困难的问题。

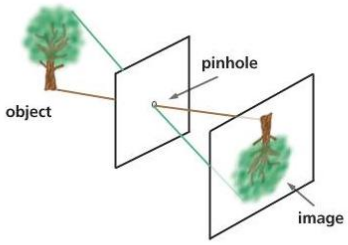
就学习 OpenMV 的角度上来讲,熟悉 Python 基本语法,各种数据结构,控制流语句就可以进行 OpenMV 的学习和使用了。

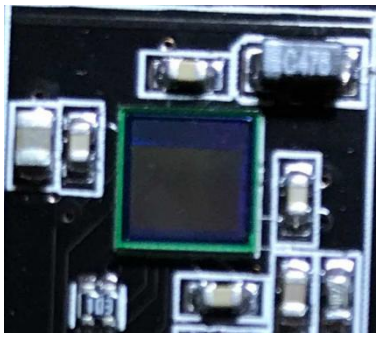
MicroPython：https://micropython.org/。

MicroPython 的库函数,随着各种发行版的使用会有所不同。OpenMV 的硬件资源控制部分,参阅文档。OpenMV-MicroPython Libraries：http://docs.openmv.io/library/index.html#python-standard-libraries-and-micro-libraries。

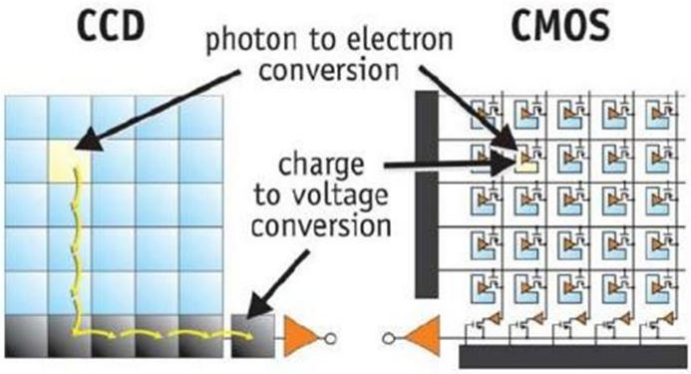
18.3 机器视觉常识

摄像头分为数字摄像头和模拟摄像头两大类。数字摄像头可以将视频采集设备产生的模拟视频信号转换成数字信号，进而将其储存在计算机里。模拟摄像头捕捉到的视频信号必须经过特定的视频捕捉卡将模拟信号转换成数字模式，并加以压缩后才可以转换到计算机上运用。数字摄像头可以直接捕捉影像，然后通过串、并口或者 USB 接口传到计算机里。景物=>光学图像=>电学信号=>数字图像信号=>PC 显示景物通过镜头产生光学图像；光学图像再同半导体的图像传感器生成电学信号；电学信号由 A/D 转换器转化为数字图像信号；数字图像信号经由 DSP 处理，在 USB 连接下在 PC 上显示出来。说到底，摄像头就是一个将光学信号转变成电信号的一个装置。在计算机视觉中，最简单的相机模型是小孔成像模型如图所示：





感光芯片（CCD or CMOS die）：

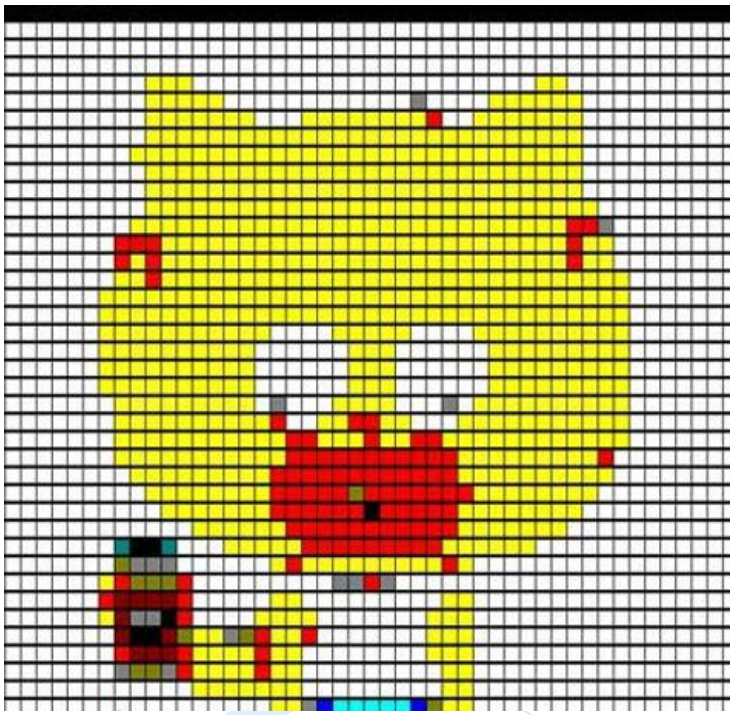


CCD(Charge Coupled Device)电荷耦合组件，用于录像或者图像扫描；灵敏度高，噪声小，信噪比大，但成本高，生产工艺复杂，功耗高。

CMOS(Complementary Metal-Cxide Semiconductor)附加金属氧化物半导体组件，是低端视频设备；集成度高，功耗低(不到 CCD 的 1/3)，成本低，但是噪声大，灵敏度低，对光源要求高。

什么是像素和分辨率

像素，为视频显示的基本单位，译自英文“pixel”，pix 是英语单词 picture 的常用简写，有时亦被称为 pel（picture element）。在很多情况下，它们采用点或者方块显示。每个像素可有各自的颜色值，可采三原色显示，因而又分成红、绿、蓝三种子像素（RGB 色域），或者青、品红、黄和黑（CMYK 色域，印刷行业以及打印机中常见）。照片是一个个取样点的集合，在视频没有经过不正确的/有损的压缩或相机镜头合适的前提下，单位面积内的像素越多代表分辨率越高，所显示的视频就会接近于真实物体。比如有 640*480 个点，每个点就是一个像素，把每个点的像素收集整理起来，就是一副图片，那么这张图片的分辨率就是 640*480：



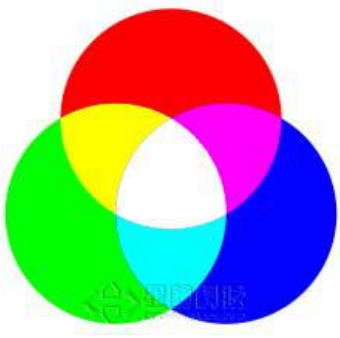
什么是帧率

帧率（FPS）就是每秒钟处理的图片数量，如果超过 20 帧，人眼就基本分辨不出卡顿。当然，如果用在机器上，帧率是越高越好的。

注：没有标注均为不传输图像给 IDE，因为这个过程很耗费时间。

Openmv 颜色空间（RGB 颜色空间和 LAB 颜色空间）

RGB 三原色的原理不是物理原因，而是由于人的生理原因造成的。人的眼睛内有几种辨别颜色的锥形感光细胞，分别对黄绿色、绿色和蓝紫色（或称紫罗兰色）的光最敏感（波长分别为 564、534 和 420 纳米）。



Lab 颜色空间中，L 亮度；a 的正数代表红色，负端代表绿色；b 的正数代表黄色，负端代表兰色。不像 RGB 和 CMYK 色彩空间，Lab 颜色被设计来接近人类视觉。因此 L 分量可以调整亮度对，修改 a 和 b 分量的输出色阶来做精确的颜色平衡。

其他颜色模型参考：

[小波的世界]HSI 颜色空间及其应用

http://nkwavelet.blog.163.com/blog/static/227756038201471975037 22

[百度文库] RGB、Lab、YUV、HSI、HSV 等颜色空间的区别

https://wenku.baidu.com/view/f38c04e69b89680203d82513.html

图像的整体属性

高度 height img.width()

宽度 width img.width()

图像的大小 img.size(),size 的取值其实是 width * height,就是说一共有多少个 pixels

图像格式：img.format()，图像格式一共有两种：sensor.GRAYSCALE:8-bits per/ pixel.sensor.RGB565:16-bits per pixel.

函数返回数值为整数，取值范围是 0/1：format0: 灰度图，每个 pixel 占 8 个位 format1:RGB 图，每个 pixel 占 16 个位。

此时你已心生疑虑，对于 grayscale 是单个数值，占 8 个位

rgb 应该是其三倍啊，因为其存储了三个值，但是为啥是 16 个位呢详情参见 RGB565 详解

实验代码：

```
import sensor, image, time
sensor.reset()      # Reset and initialize the sensor.
sensor.set_pixformat(sensor.RGB565) # Set pixel format to RGB565 (or GRAYSCALE) sensor.set_framesize(sensor.QVGA)
clock = time.clock()    # Create a clock object to track the FPS. while(True):
clock.tick()    # Update the FPS clock.
img = sensor.snapshot() # Take a picture and return the image. print("IMG Format")
print(img.format())
print("IMG Width")
print(img.width())
print("IMG Height")
print(img.height())
print("IMG SIZE")
print(img.size())
```

样例输出

IMG Format 1

IMG Width 320

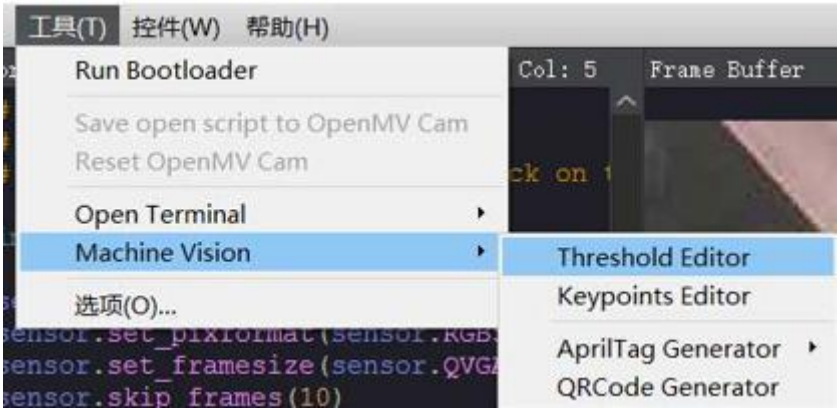
IMG Height 240

IMG SIZE 153600

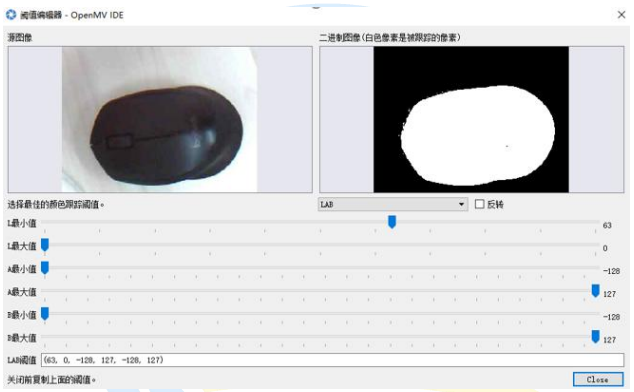
阈值选择工具

一个颜色阈值的结构是这样的：red = (minL, maxL, minA, maxA, minB, maxB)元组里面的数值分别是 L A B 的最大值和最小值。在新版的 IDE，有更方便的阈值选择工

具，打开工具→Mechine Vision→Threshold Editor 见下面。

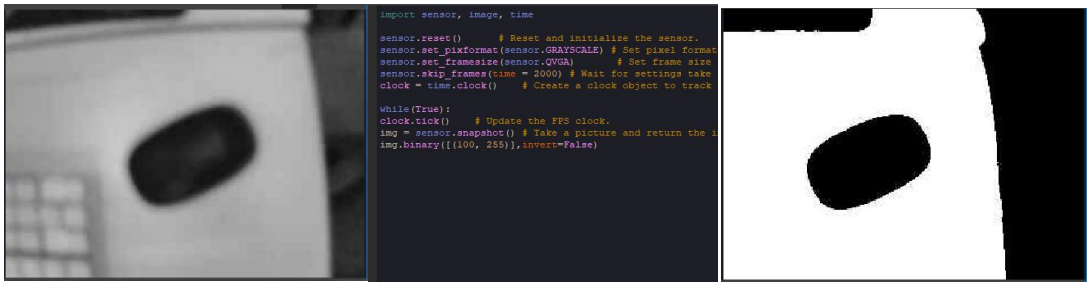


点击 Frame Buffer 可以获取 IDE 中的图像，Image File 可以自己选择一个图像文件。拖动六个滑块，可以实时的看到阈值的结果，我们想要的结果就是将我们的目标颜色变成白色，其他颜色全变为黑色。



二值化 binary

image.binary(thresholds,invert=False)thresholds 阈值为一个数组[...]，数组里面有若干个 tuple 组成， 每个 tuple 都由最大值与最小值组成，(lower,upper),即上界下界，如果是 RGB 格式的话，则需要设定六个阈值(l_lo,l_hi,a_lo,a_hi,b_lo,b_hi)，在阈值范围内的就设定为 1(white)，不再阈值范围内的就设定为 0(black)，原图、实验效果及实验代码：



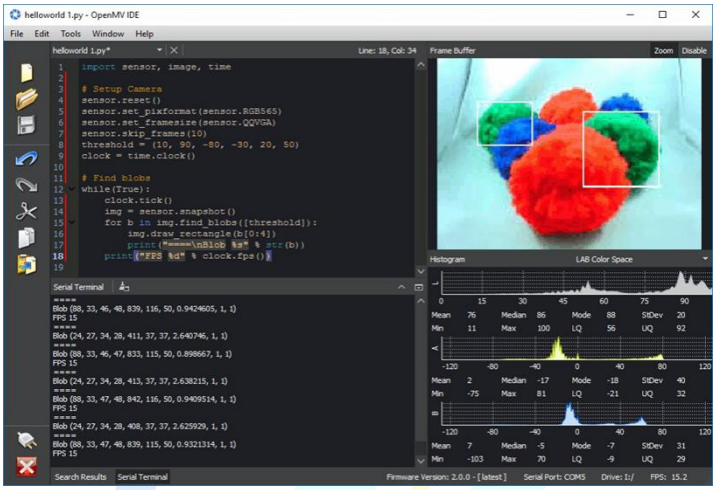
```
import sensor, image, time

sensor.reset()      # Reset and initialize the sensor.
sensor.set_pixformat(sensor.GRAYSCALE) # Set pixel format
sensor.set_framesize(sensor.QVGA)      # Set frame size
sensor.skip_frames(time = 2000) # Wait for settings take
clock = time.clock()    # Create a clock object to track

while(True):
clock.tick()    # Update the FPS clock.
img = sensor.snapshot() # Take a picture and return the i
img.binary([(100, 255)],invert=False)
```

image.invert()反色就是说，1(白色)-->0(黑色)，0(黑色)-->1(白色)，相当于逻辑中的取反操作，等同于将上面图片黑色部分变成白色，白色部分变成黑色。

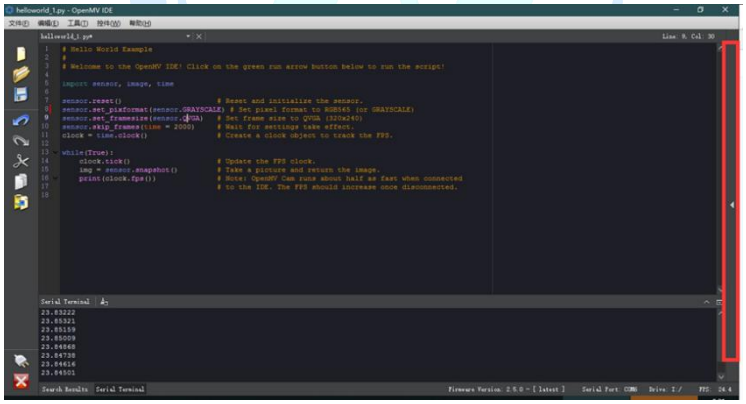
18.4 Openmv 快速上手

18.4.1 OpenMV IDE、驱动安装及使用教程

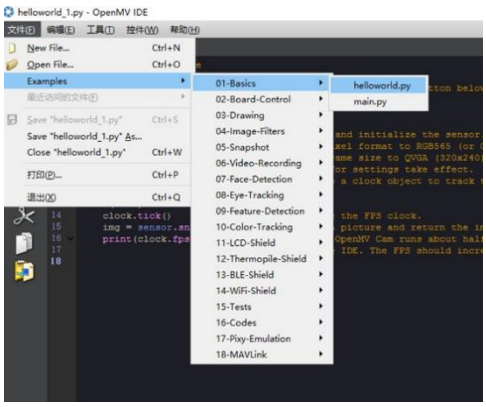
下载 IDE：https://openmv.io/pages/download，当前版本为 v1.7.1 版本，根据你操作系统的型号选择合适的安装包。因为基于 QT 所以对 Linux 也有很好的支持。IDE 工作台如下图所示。



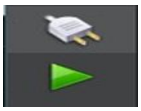
如果没有图像窗口，从右侧可以拖拽出来。



查看样例代码：

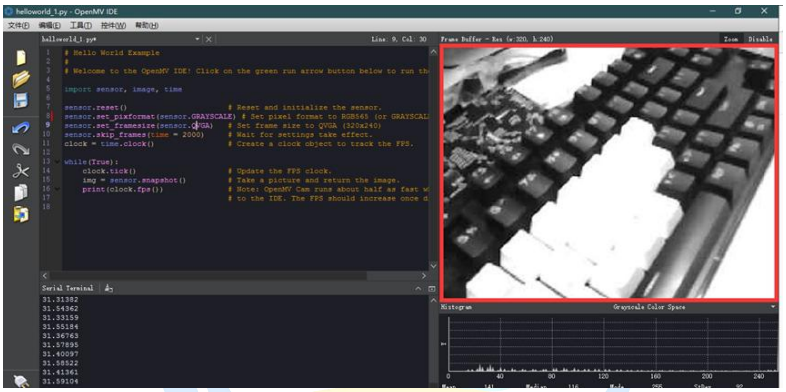


串口连接与代码烧录

点击链接串口，就是左下角那个白色的按钮  连接后高亮，并且显示绿色的按钮，说明可以运行代码了  点击绿色的按钮，代码上传给 OpenMV，程序执行，这时在视频区域你可以看到事实的图像了。此时按钮会变成红色的 X，如果要中断程序，或者改动了代码需要更新，点击 X，然后再重新运行。

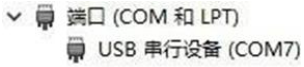
视频显示

右上角显示了实时的图像信息，如果你想截图取样的话，直接左键选中一个矩形区域然后右键保存到本地，选择合适的格式。



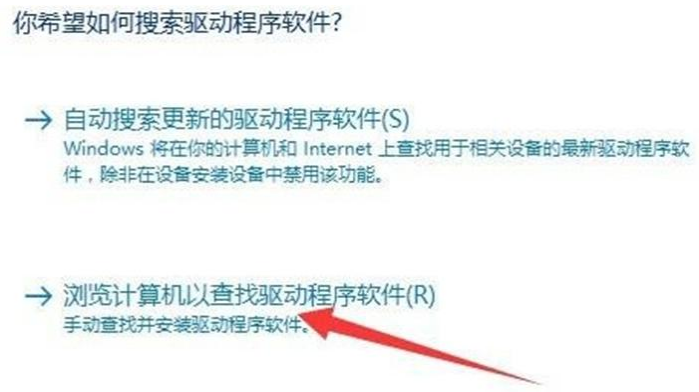
驱动安装

正常情况下，OpenMV 插入电脑上，会自动安装驱动。目前正常工作在 32 位和 64 位的 win7、win8、win8.1、win10。自动安装后，在设备管理器会出现一个虚拟串口。

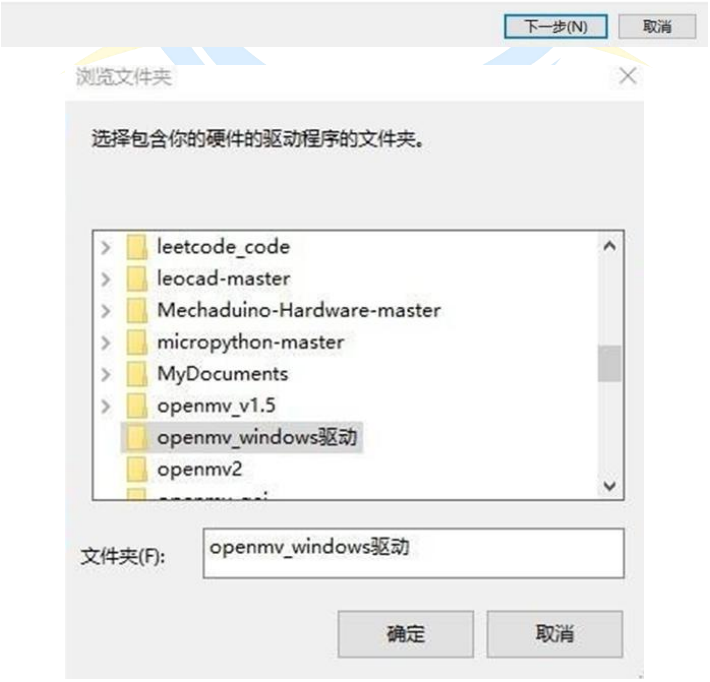


，但是可能驱动不会自动安装，所以需要自己手动安装。这时在设备管理器中会出现一个叹号，表示没有正常安装驱动。

首先下载驱动：http://pan.baidu.com/s/1pKQd59L 解压到桌面，然后右键设备管理器中的这个设备，然后点升级驱动：



选择浏览计算机以查找驱动程序软件



选中桌面上的 openmv_windows 驱动文件夹，过一会就会正常装好驱动。

这时应该会成功的安装好驱动。但是 但是 但是，如果你遇到下面的情况，就比较麻烦了。

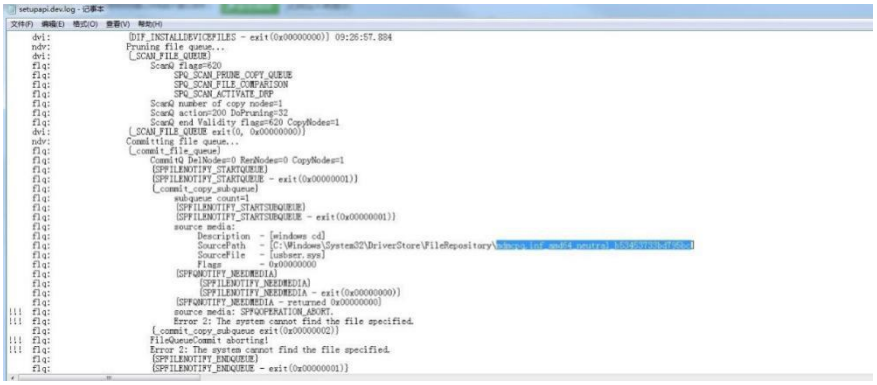
这是因为有些系统用了一些精简系统的软件，把一些不常用的驱动删除了，但是 OpenMV 的驱动依赖这些驱动，所以要把他们粘贴回来。

驱动安装失败解决办法

OpenMv 驱动安装失败，90%的情况都是你电脑的问题，精简版操作系统和使用了一些优化软件通常是引起此类问题的原因。OpenMV 驱动解决办法跟 arduino 类似。这是因为精简版的 window 系统删掉了一些不常用的驱动信息引起的，解决方法如下：

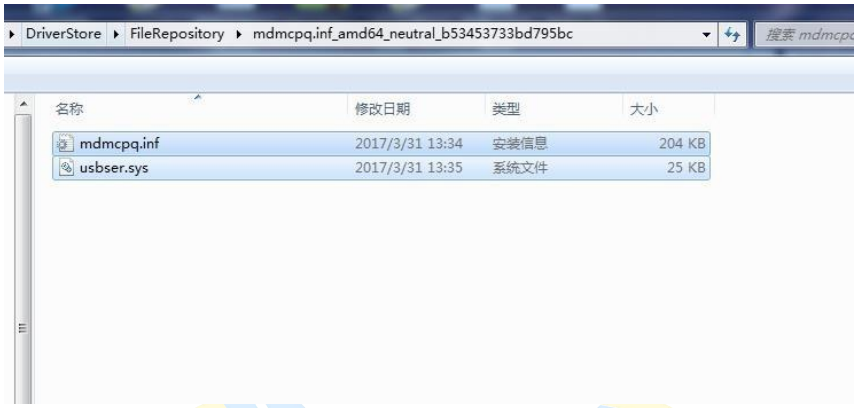
Step1 首先打开 C:\windows\inf\setupapi.dev.log

这个文件包含了有关即插即用设备和驱动程序安装的信息，当然它也记录你 Arduino 驱动安装失败的原因。打开该文件，滚动到文件末尾附近，你可以看到如下信息：



Step2 在 C:\Windows\System32\DriverStore\FileRepository\路径下，新建一个 mdmcpq.inf_amd64_neutral_b53453733bd795bc（这个根据你电脑的提示修改）文件夹

Step3 将丢失文件拷贝到刚新建的文件下后，从新安装驱动，不出意外可以安装成功，缺失驱动文件在网盘中下载 http://pan.baidu.com/s/1jIxfxoM



18.4.2 固件编译、烧录、升级

编译固件及固件升级：https://github.com/openmv/openmv/wiki 固件的烧录：https://singtown.com/video/

18.4.3 使用注意事项

OpenMV3 拔插注意事项，OpenMV3 的存储有两种，一种是 Flash，一种是 SD 卡。 如果需要频繁的烧录程序， 建议配备一个 SD 卡，将程序存放在 SD 卡中。默认是 Flash 在 OpenMV3 拔下来之前需要断开 OpenMV 的串口连接，弹出 Flash 存储。注意！尽量别在程序运行的时候按复位键， 另外,Flash 易损,平常使用过程中要注意避免热拔插。

18.5 常用例程

18.5.1 基本操作

```
import sensor, image, time

sensor.reset()                      # Reset and initialize the sensor.
sensor.set_pixformat(sensor.RGB565) # Set pixel format to RGB565 (or GRAYSCALE)
sensor.set_framesize(sensor.QVGA)   # Set frame size to QVGA (320x240)
sensor.skip_frames(time = 2000)     # Wait for settings take effect.
clock = time.clock()                # Create a clock object to track the FPS.

while(True):
    clock.tick()                    # Update the FPS clock.
    img = sensor.snapshot()         # Take a picture and return the image.
    print(clock.fps())              # Note: OpenMV Cam runs about half as fast when connected
                                    # to the IDE. The FPS should increase once disconnected.
```

用 usb 线与电脑连接后，打开文件——examples——01Basic——helloworld.py 例程，点击左下角绿色箭头按钮运行。import sensor, image, time 引入此例程依赖的模块，sensor 是与摄像头参数设置相关的模块，image 是图像处理相关的模块，time.clock 时钟控制相关的模块。import 相当于 c 语言的#include<>，模块相当于 c 语言的库。sensor.reset()#初始化摄像头，reset()是 sensor 模块里面的函数；sensor.set_pixformat(sensor.GRAYSCALE)设置图像色彩格式，有 RGB565 色彩图和 GRAYSCALE 灰度图两种；sensor.set_framesize(sensor.QVGA).设置图像像素大小，sensor.QQVGA:160x120，sensor.QQQVGA:80x60，sensor.QQVGA2:128x160 (一般用于 LCD 扩展板)，sensor.QVGA:320x240，sensor.QQCIF:88x72，sensor.QCIF:176x144，sensor.CIF: 352x288，sensor.skip_frames(10)，clock =time.clock()初始化时钟,注意：python 中没有大括号｛｝，以缩进代替｛｝，而且缩进的 tab 和空格不能混用，一个程序只能用一种缩进（一个 tab 或者四个空格）

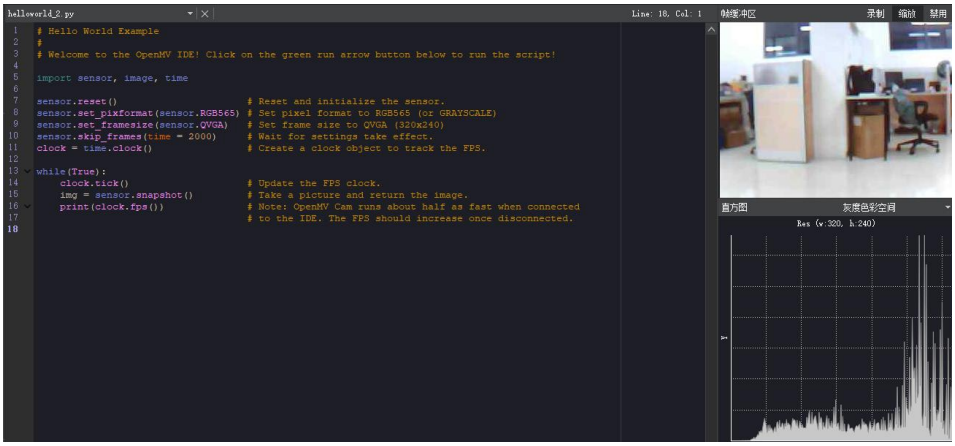
while(True):

#python while 循环，一定不要忘记加冒号“:”

clock.tick()

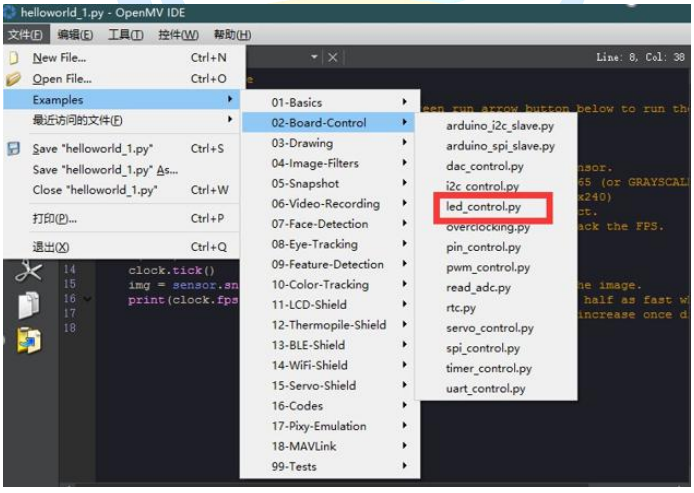
img = sensor.snapshot() # Take a picture and return the image.

#截取当前图像，存放于变量 img 中。注意 python 中的变量是动态类型，不需要声明定义，直接用即可。print(clock.fps())#打印当前的帧率。然后就可以看到运行效果啦。



18.5.2 点亮 LED 灯

OpenMV 摄像头有一个 RGB LED 和两个红外 LED 灯。您可以单独控制 RGB LED 的红色，绿色和蓝色区段，将两个 IR LED 控制为一个单位。要控制 LED，首先导入 pyb 模块。然后为要控制的特定 LED 创建一个 LED 类对象，pyb.LED(number)创建一个 LED 对象，您可以使用它来控制特定的 LED。通过 pyb.LED0”控制红色 RGB LED，“1”控制绿色 RGB LED，“2”控制蓝色 RGB LED，“3”控制两个红外 LED。在创建如上所述的 LED 控制对象后，有三种方法可以调用每个 LED 有 off(),on(),与 toggle()三个选项。按照目录打开此文件 led_control.py。

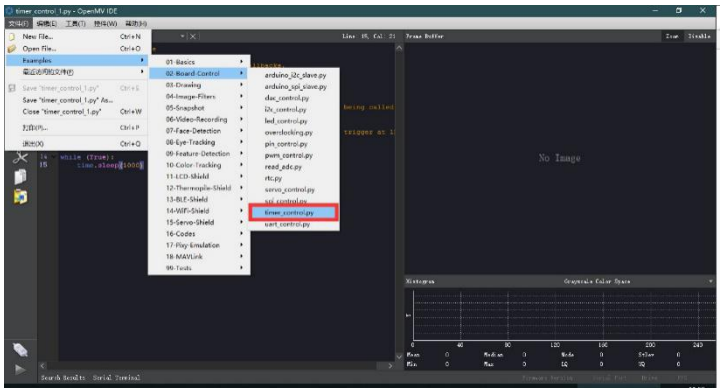


18.5.3 定时器的使用

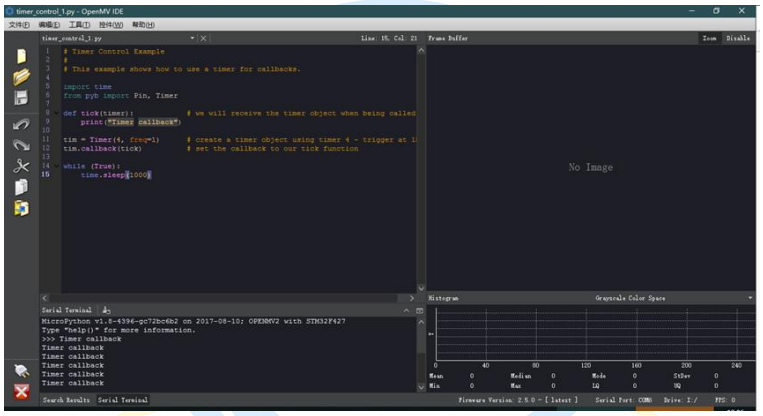
官方文档:http://docs.openmv.io/library/pyb.Timer.html?highlight=timer

意思也就是 Timer1 用于控制摄像头，Timer5 用于控制舵机驱动，Timer6 用于定 ADC/DAC 读或写，所以，我们编程的时候不能使用这三个。那也就是说，只可以用定时器 2,

3,4，当然，没用到那些外设的时候，也就是只用 LED 的时候 14 个定时器都能用。打开图中所示文件。



此程序初始化了定时器 4，频率设置为 1hz，回调函数为 tick，也就是每隔 1s 进一次 tick 函数，打印一句 Timer callback，效果如下。



注意：定时器调用的时候，不可以开辟新的内存，这一点很重要，当你在定时器的回调函数定义一个新的变量或者定义一个字符串的时候，就会提示有 Memory Error 的错误。同时也意味着如果定时器函数回调期间出现了 BUG，不能返回完整的报错信息。所以我们需要提前申请一块缓存，用于异常处理。

```
import micropython

micropython.alloc_emergency_exception_buf(100)
```

未加紧急异常缓存的报错信息：

```
uncaught exception in Timer(4) interrupt handler
TypeError:
```

定时器的正确用法：

假定现在我们有个需求，需要使用定时器，阶段性的检查串口是否有数据，并且将其读入。如何实现？如何克服我们之前说的那个，callback 时不可以申请内存的问题？解决方法是，定时器只判断修改标志位（flag，标志位设置为全局变量 global），然后在 while 循环里，根据标志位，做相应的处理。

```
has_data = False # 初始化为没有数据读入

def is_data_coming(timer):
    global uart
    global has_data

    if uart.any():
        has_data = True
    else:
        has_data = False


timer = Timer(4)
timer.init(freq=10)
timer.callback(led_on_off)


while True:
    # 这里只延时 啥也不干
    if has_data:
        data = uart.readline()
        print(data)
```

18.5.4 UART 串口通信

更全面请看:http://docs.micropython.org/en/latest/pyboard/library/pyb.UART.html?highlight=uart

读写操作：

uart.read(10)：读入 10 个字符，返回一个比特对象

uart.read()：读取所有的有效字符

uart.readline()：读入一行

uart.readinto(buf)：读入并且保存在缓存中

uart.write('abc')：向串口写入 3 个字符 abc

单个字符的读取与写入：

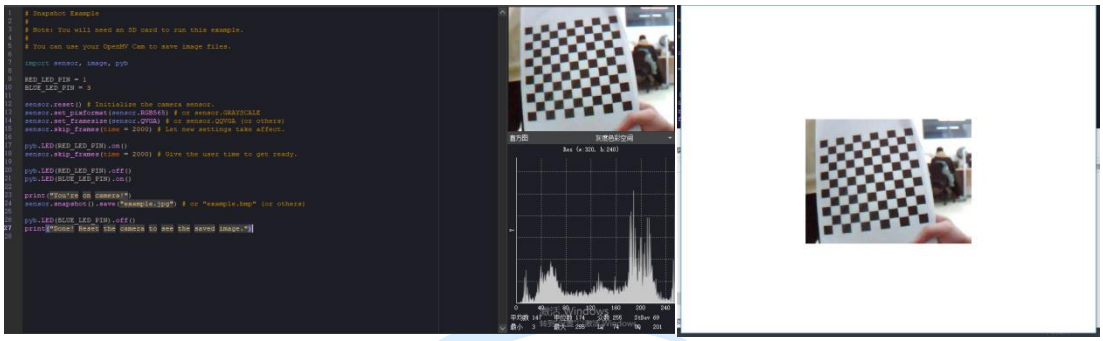
uart.readchar()：读入一个字符

uart.writechar(42)：写入 ASCALL 码为 42 的字符

uart.any()：判断串口是否有数据

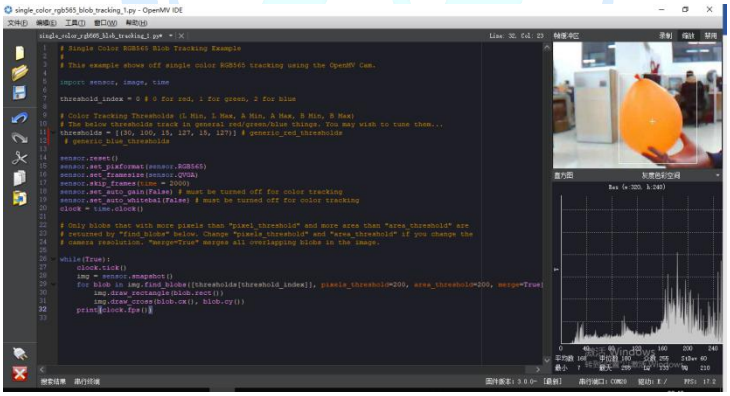
18.5.5 拍照

当我们要进行摄像头参数的标定时，我们需要使用 openmv 拍摄棋盘图，故使用的例程为 Snapshot-snapshot.py，本例程的目标是使用 save 函数保存摄像头图片。注意：因为 openmv 内存较小，需要外接 SD 卡才能保存图片哦。



18.5.6 颜色追踪

本例程为 10-Color_Ttracking-blob_detection 本例程的目标是用 OpenMV 实现颜色识别。openmv 可以多个颜色同时识别。



例如：green_threshold = (0, 80, -70, -10, -0, 30)设置绿色的阈值，括号里面的数值分别是 L A B 的最大值和最小值（minL, maxL, minA, maxA, minB, maxB），LAB 的值在图像左侧三个坐标图中选取。如果是灰度图，则只需设置（min, max）两个数字即可。

```
sensor.reset()
sensor.set_pixformat(sensor.RGB565)
sensor.set_framesize(sensor.QQVGA)
sensor.skip_frames(10)
```

sensor.set_auto_whitebal(False) #关闭白平衡。白平衡是默认开启的，在颜色识别中，需要关闭白平衡。

```
clock = time.clock()
  while(True):
    clock.tick()
    img = sensor.snapshot()
    blobs = img.find_blobs([green_threshold]) #find_blobs(thresholds, invert=False, roi=Auto),thresholds 为颜色阈值，是一个元组，需要用括号 [] 括起来。invert=1,反转颜色阈值，invert=False 默认不反转。roi 设置颜色识别的视野区域，roi 是一个元组，roi = (x, y, w, h)，代表从左上顶点(x,y)开始的宽为 w 高为 h 的矩形区域，roi 不设置的话默认为整个图像视野。这个函数返回一个列表，[0]代表识别到的目标颜色区域左上顶点的 x 坐标，[1]代表左上顶点 y 坐标，[2 代表目标区域的宽，[3]代表目标区域的高，[4]代表目标区域像素点的个数，[5]代表目标区域的中心点 x 坐标，[6]代表目标区域中心点 y 坐标，[7]代表目标颜色区域的旋转角度（是弧度值，浮点型，列表其他元素是整型），[8]代表与此目标区域交叉的目标个数,[9]代表颜色的编号(它可以用来分辨这个区域是用哪个颜色阈值 threshold 识别出来的）。
    if blobs:如果找到了目标颜色
        for b in blobs:迭代找到的目标颜色区域
            img.draw_rectangle(b[0:4])用矩形标记出目标颜色区域
            img.draw_cross(b[5], b[6])在目标颜色区域的中心画十字形标记
            print(b[5], b[6])输出目标物体中心坐标
  print(clock.fps())
```

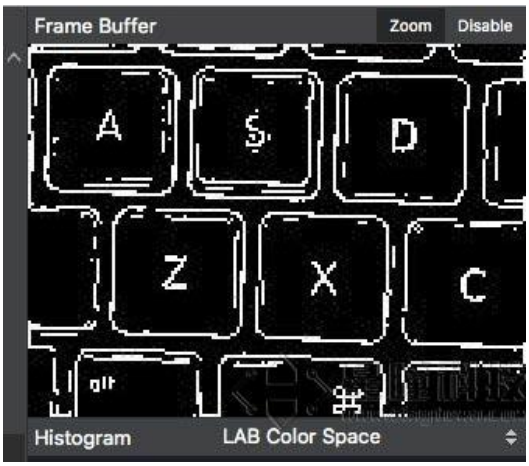
18.5.7 腐蚀与膨胀

腐蚀操作首先对图像进行分割，二值化，将在阈值内的区域变为白色，阈值外区域变为黑色，再对图像边缘进行侵蚀，侵蚀函数 erode(size, threshold=Auto)，size 为 kernal 的大小，去除边缘相邻处多余的点。threshold 用来设置去除相邻点的个数，threshold 数值越大，被侵蚀掉的边缘点越多，边缘旁边白色杂点少；数值越小，被侵蚀掉的边缘点越少，边缘旁边的白色杂点越多。

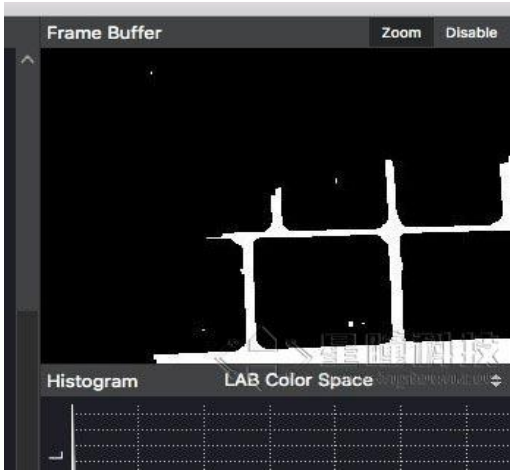
膨胀操作对图像边缘进行膨胀，膨胀函数 image.dilate(size, threshold=Auto)，size 为 kernal 的大小，使边缘膨胀。threshold 用来设置去除相邻点的个数，threshold 数值越大，边缘越膨胀；数值越小，边缘膨胀的小。具体代码如下所示。

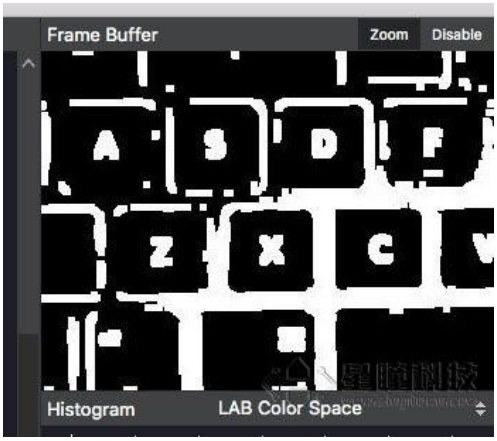
```
while(True):
    sensor.set_pixformat(sensor.GRAYSCALE)
    for i in range(20):
        img = sensor.snapshot()
        img.binary([grayscale_thres])
    for i in range(20):
        img = sensor.snapshot()
        img.binary([grayscale_thres])
        img.dilate(2)
    sensor.set_pixformat(sensor.RGB565)
    for i in range(20):
        img = sensor.snapshot()
        img.binary([rgb565_thres])
        img.erode(2)
    for i in range(20):
        img = sensor.snapshot()
        img.binary([rgb565_thres])
        img.dilate(2)
```

实现效果如下所示。



原图　　边缘检测后的原图





erode 函数腐蚀后的原图　　dilate 函数膨胀后的原图

膨胀与腐蚀

http://blog.csdn.net/poem_qianmo/article/details/23710721

开运算、闭运算、形态学梯度、顶帽、黑帽合辑

http://blog.csdn.net/poem_qianmo/article/details/24599073

18.5.8 条形码和二维码检测

条形码检测可以在 OpenMV Cam 的 OV7725 相机模块的 640x480 分辨率下运行。条码检测也将在 RGB565 模式下工作，但分辨率较低。也就是说，条形码检测需要更高的分辨率才能正常工作，因此应始终以 640x480 的灰度运行：

http://book.openmv.cc/example/16-Codes/find-barcodes.html

二维码检测：

http://book.openmv.cc/example/16-Codes/qrcodes-with-lens-zoom.html

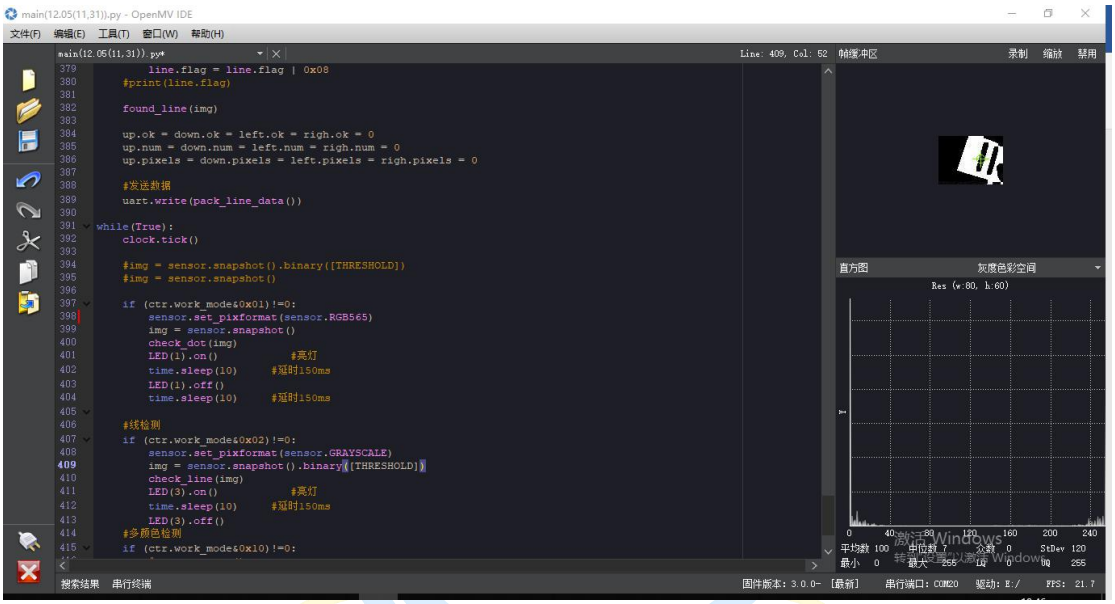
18.6 电赛教程

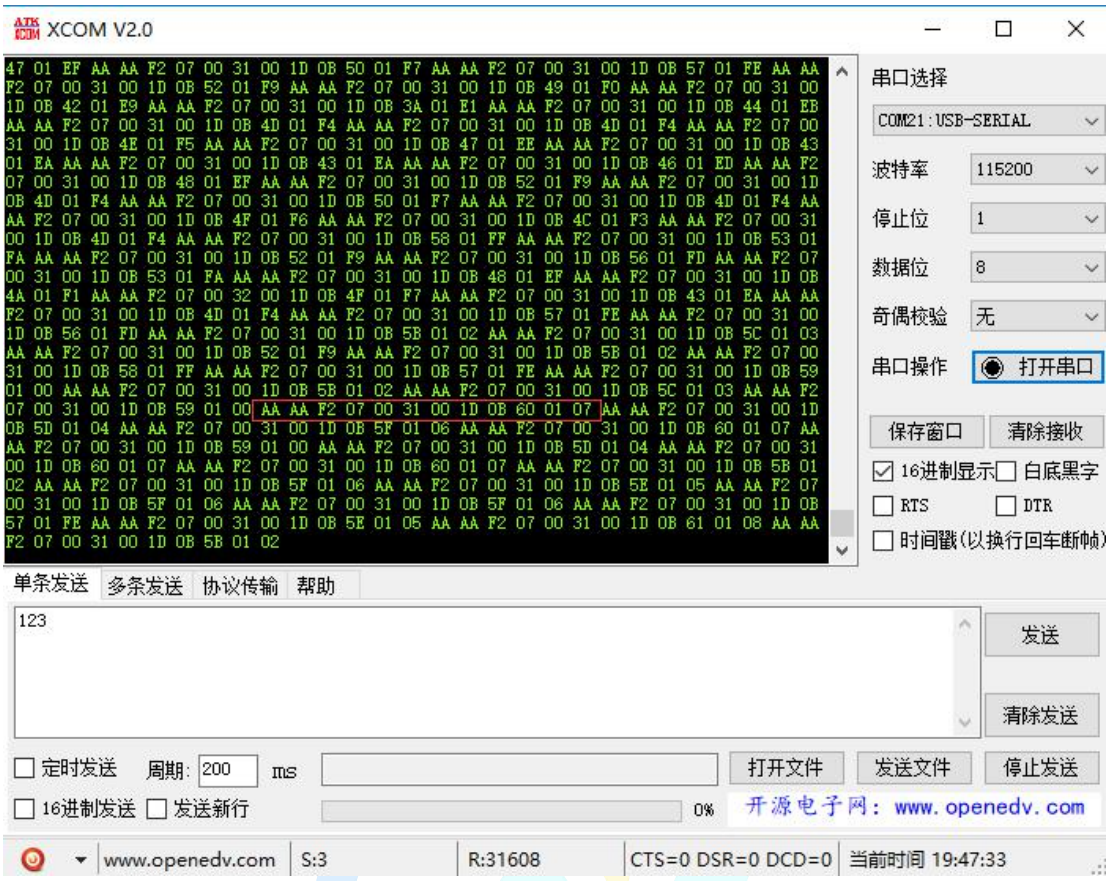
18.6.1 黑点或色块检测

色块坐标输出思路：先初始化摄像头等传感器模块，然后利用前面介绍的阈值编辑器调节需要识别色块的阈值大小，再调用用 img.find_blobs（）查找色块，找到不同的色块后，使用 dot.pixels<blob.pixels()比较得出最大的色块像素点，最后调用 blob.cx()，blob.cy()得出 x,y 的坐标，将得到的坐标值减去整个图像的中点像素坐标

值即可得到需要色块的相对于图像中心的坐标偏差值。再用 uart.write(pack_dot_data())（串口发送详见代码部分）函数发送黑点或者色块的坐标，部分代码如下：

```
#点检测函数
def check_dot(img):
    for blob in img.find_blobs(thresholds, pixels_threshold=150, area_threshold=150, merge=True, margin=5):
        if dot.pixels<blob.pixels():#寻找最大的黑点
            img.binary(thresholds)
            img.erode(2)
            dot.pixels=blob.pixels()
            dot.x = blob.cx()
            dot.y = blob.cy()
            dot.ok= 1
            img.draw_cross(dot.x, dot.y, color=127, size = 10)
            img.draw_circle(dot.x, dot.y, 5, color = 127)
```





点检测数据解读：

Bit1	AA	帧头
Bit2	AF	帧头
Bit3	F2	点数据标志位
Bit4	07	有效数据位
Bit5	00	Dot.x 坐标高八位
Bit6	3B	Dot.x 坐标低八位
Bit7	00	Dot.y 坐标高八位
Bit8	06	Dot.x 坐标低八位
Bit9	0C	像素数高八位
Bit10	F2	像素数低八位
Bit11	01	点数据标志位（有数据为 1，没有数据为 0）
Bit12	00	保留

1. dot_x ：点的 X 坐标（数据范围：0-80，单位：像素点）。

2.dot_y ： 点的 Y 坐标（数据范围：0-40，单位：像素点）。

3. flag：是否检测到点（1：是，0：否）。

18.6.2 无人机巡线

巡线部分使用霍夫变换进行线段的拟合，然后将图像部分分成 4 个 roi 区域，根据标志位来判断直线在每个区域的位置，来判断线段是一条线还是两条，是横线还是竖线。如果是两条线，是十字相交还是 T 字相交或者是 7 字相交。霍夫换巡线如下所示。

```
def found_line(img):
    singleline_check.flager = img.get_regression([(255,255)], robust = True)
    if (singleline_check.flager):
        #print(clock.fps())
        singleline_check.rho_err = abs(singleline_check.flager.rho())-0
        if singleline_check.flager.theta()>90:
            singleline_check.theta_err = singleline_check.flager.theta()-0
        else:
            singleline_check.theta_err = singleline_check.flager.theta()-0
        img.draw_line(singleline_check.flager.line(), color = 127)
        print(singleline_check.theta_err)


def check_line(img):
    fine_border(img,up,up_roi)
    fine_border(img,down,down_roi)
    fine_border(img,left,left_roi)
    fine_border(img,righ,righ_roi)

    line.flag = 0
    if up.ok:
        line.flag = line.flag | 0x01
    if down.ok:
        line.flag = line.flag | 0x02
    if left.ok:
        line.flag = line.flag | 0x04
    if righ.ok:
        line.flag = line.flag | 0x08
    #print(line.flag)

    found_line(img)

    up.ok = down.ok = left.ok = righ.ok = 0
    up.num = down.num = left.num = righ.num = 0
    up.pixels = down.pixels = left.pixels = righ.pixels = 0

    #发送数据
    uart.write(pack_line_data())
```

4 个 roi 区域的划分图如下所示：

线检测数据解读：

Bit1	AA	帧头
Bit2	AF	帧头
Bit3	F3	线数据标志位
Bit4	07	有效数据位

Bit5	00	singleline_check.rho_err 高八位
Bit6	3B	singleline_check.rho_err 低八位
Bit7	00	singleline_check.theta_err 高八位
Bit8	06	singleline_check.theta_err 低八位
Bit9	01	线数据标志位（有数据为 1，没有数据为 0）
Bit10	F2	校验和
Bit11	00	保留
Bit12	00	保留

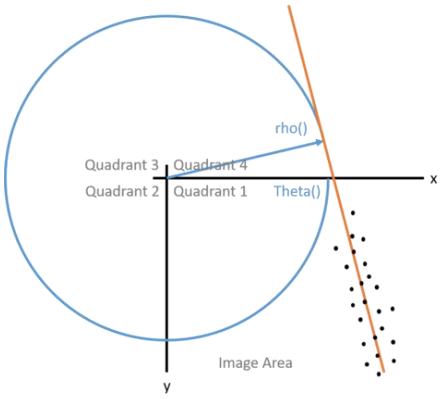
Flag 标志位各位描述：

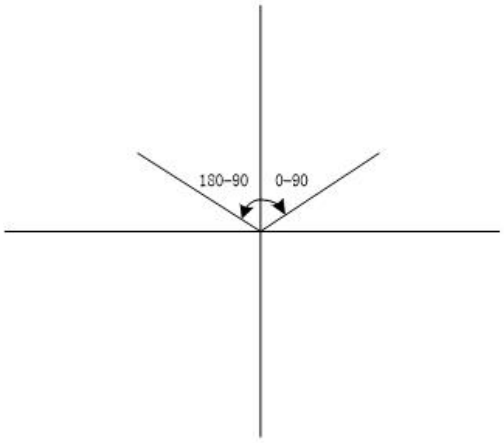
bit7-4	保留
bit3	1： 有线穿过视窗右边 0： 没线穿过视窗右边
bit2	1： 有线穿过视窗左边 0： 没线穿过视窗左边
Bit1	1： 有线穿过视窗下边 0： 没线穿过视窗下边
Bit0	1： 有线穿过视窗上边 0： 没线穿过视窗上边

线检测数据解读：

1、 singleline_check.rho_err：线偏离中线的偏差（数据范围：0-80，单位像素点）。

2、 singleline_check.theta_err：线偏离中线的倾角（数据范围：0-180，单位：度）。





3、 line.flag：是否有目标线穿过视窗中上下左右四个边的标志。

由于时间紧，整理不详尽的地方还请谅解，更多飞控介绍，详见后续教学视频。

未完待续！！！