“智行高速“使用文档

本项目基于树莓派与嵌入式平台，构建了智能车牌识别控制系统，适用于高速收费站场景。

参考项目<https://github.com/l4hu/Car-Park-License-Plate-Recognition-System>

整体流程：

系统以STM32微控制器为下位机，搭配光敏传感器，精准感知车辆靠近信号。车辆接近时，下位机接收信号，触发语音模块提示音，同时经串口通信启动树莓派（上位机）车牌识别线程。该线程利用YOLO v8模型定位车牌，裁剪预处理后，结合OCR技术识别车牌文字。识别结果经授权判断模块与数据库比对，判定车辆通行权限。针对已授权车辆上位机会回传通行指令，下位机控制舵机开启道闸；针对未授权车辆则回传拒绝指令，下位机保持道闸关闭并显示收费二维码。缴费完成后，系统控制道闸升起完成放行，在车辆离开后复位。图示

描述已自动生成

附加功能：

后端集成Web交互界面，系统会同步更新车辆活动日志与通行统计数据，同时提供路口实时监控画面与车牌拍摄图片和车牌信息，提供数据库修改及手动检测功能，方便管理员远程操作。系统运行时，AHT20温湿度传感器实时感知环境，通过PWM技术自动调控风扇转速，维持舒适温度，支持物理按键手动干预，针对突发状况，也提供了人工操控道闸的应急方案。

硬件：

上位机：树莓派5，USB摄像头

树莓派就是一个小型电脑，他也需要装入系统，就比如我们电脑里装的Windows系统这种，当然树莓派也要联网（给他开热点啥的），然后操作他最好配个显示屏，可视化方便操作，如果没有显示屏也可以使用VNC软件来实现远程投屏操作（我就是），USB摄像头就随便买个适配树莓派5的即可（温馨提示：树莓派是直接上电就开机启动了，遇到问题可以尝试拔插电源，还不行再试试拔插SD卡）。

下位机：STM32F103C8T6最小系统板

TB6612电机驱动模块（16P）

OLED（0.96寸，4P，SSD1306或SSD1315驱动），

光敏传感器模块（4P）

SG90舵机

电机（这前面的都是使用的江科大的学习套件里面的模块）

JQ8900语音模块（16P）

扬声器（与语音模块一起买）

AHT20温湿度传感器模块（4P）

有源蜂鸣器（单只蜂鸣器,2P，间距三个排针的距离）

三个按键（6x6,4P）

公排母排若干，其中排针间距都是默认的2.54mm

当然得买一个ST-Link下载器，同时你的电脑得装上ST-Link的驱动。

最后我画了一块PCB板（两层）方便接线，只有孔的板，只用焊接排针、按键和蜂鸣器即可，可以直接使用Gerber原文件进行打板（推荐嘉立创）。

其中PCB板子上有丝印标注是哪个模块，同时有接线提示，比如3.3V/GM，代表的就是板子的最左边是3.3V，应该拿模块的VCC与他相连，还有RX/SMP就是板孔左边是RX，这里应该连树莓派的TX，上下也是同理，还有丝印名称是首字母大写，光敏（GM），温湿度（WSD）,舵机（DJ），直流电机（ZLDJ），语音模块（YYMK），模式切换（MSQH，风扇电机的手动和自动挡切换，默认手动，自动挡是高于27度启动一档，29度进入二档），树莓派（SMP），喇叭（LB），备用接口（BY，那四十个排针都是与系统板的四十个引脚一一对应的，方便拓展）按键就是后缀加个（AJ，比如DJAJ）

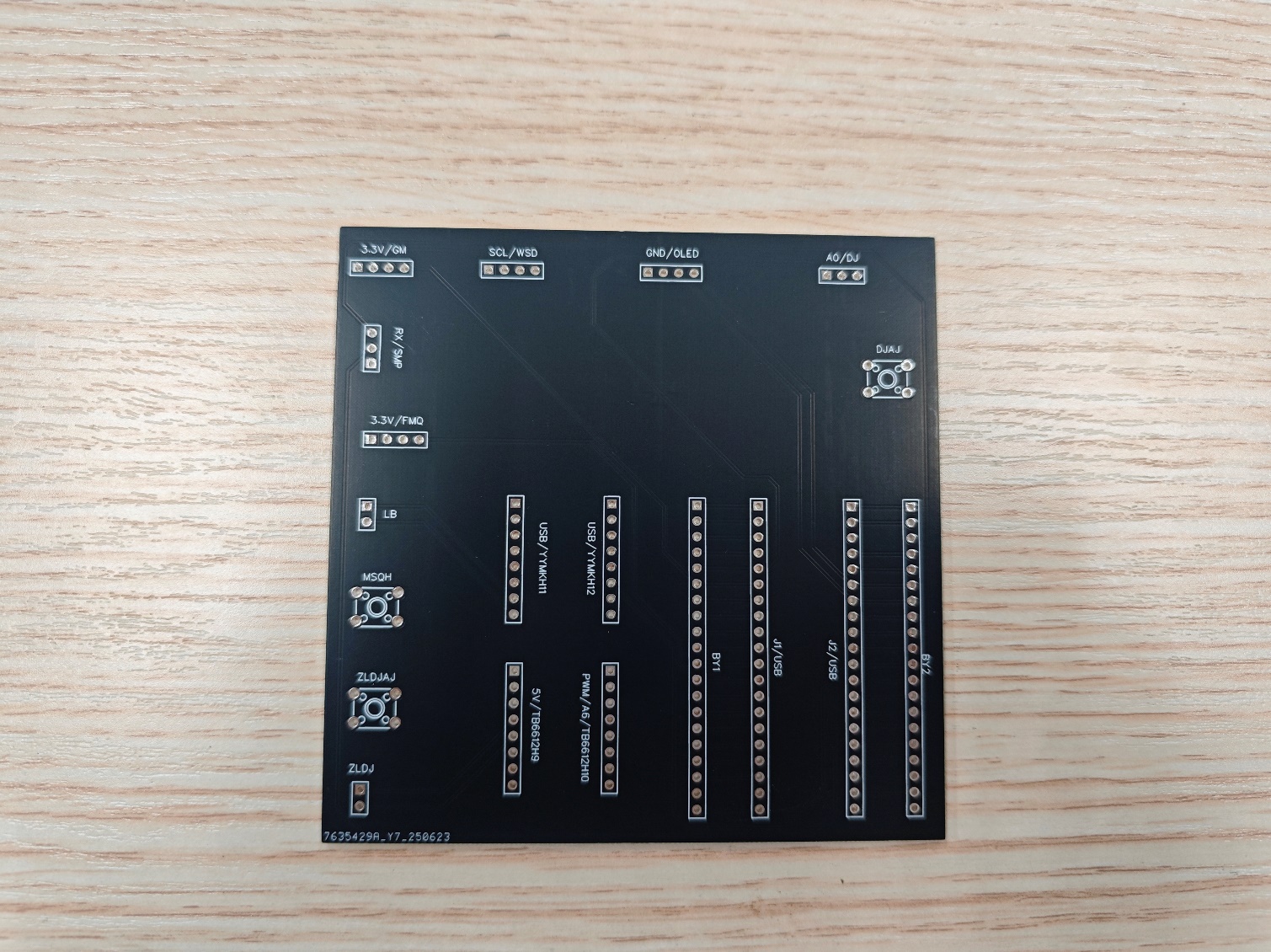


图1 PCB打板图

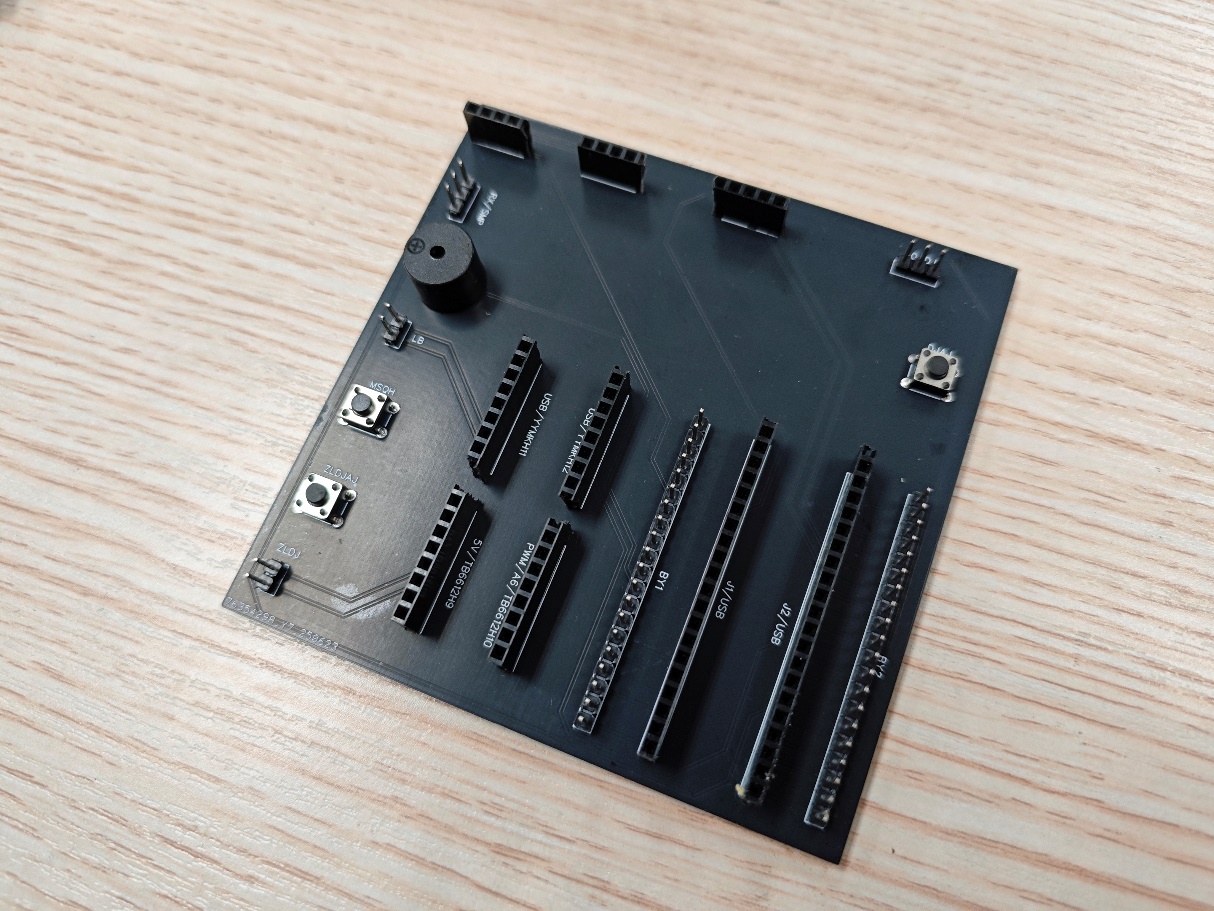


图2 焊接图

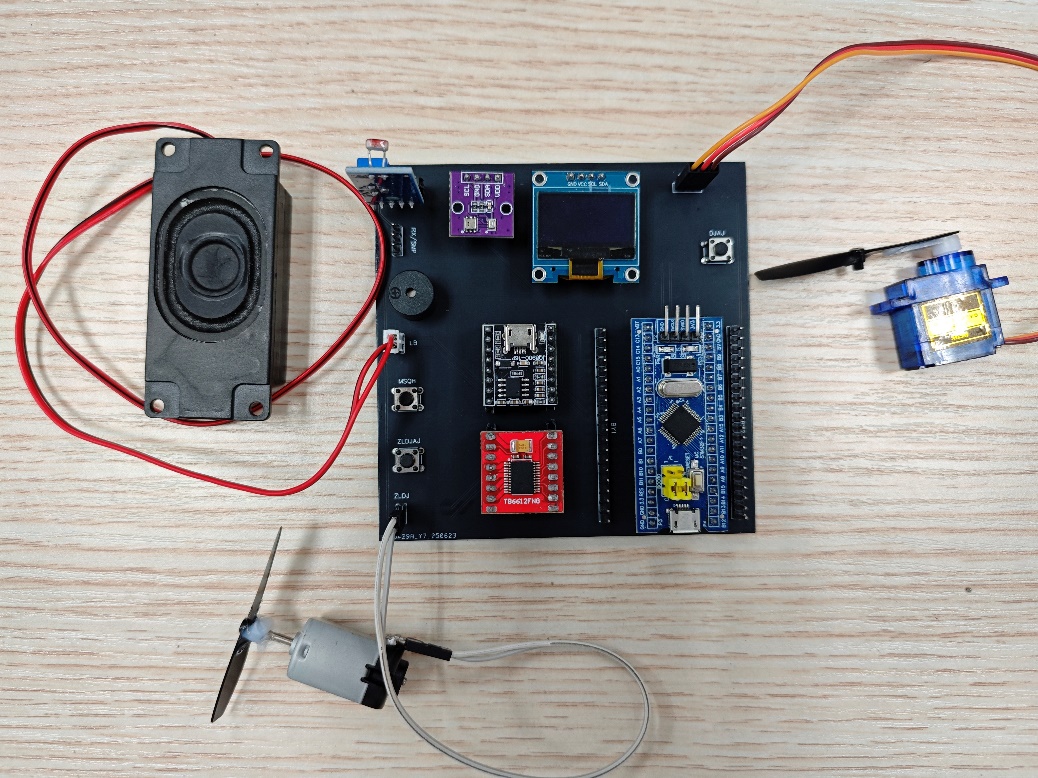


图3 下位机连接图



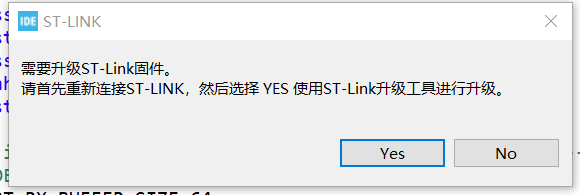
图4 上下位机完整连接图

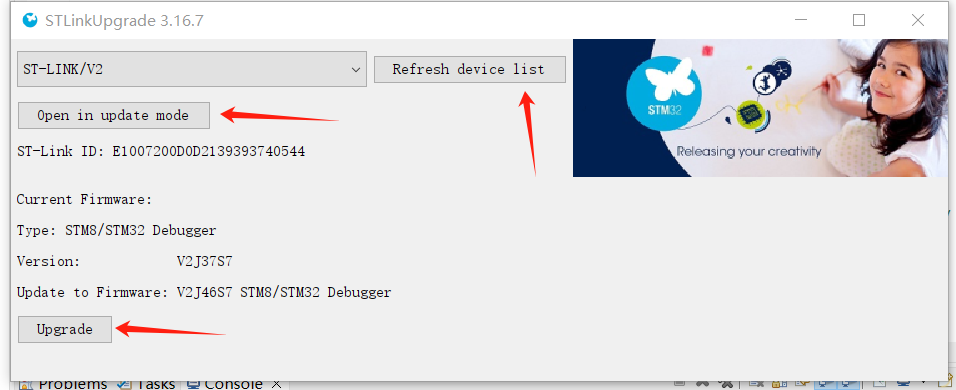
软件：

**32端的代码编译与下载：**

我使用的是CubeIDE 1.18.1版本，同时你得把F1系列的固件包下载好（标配）

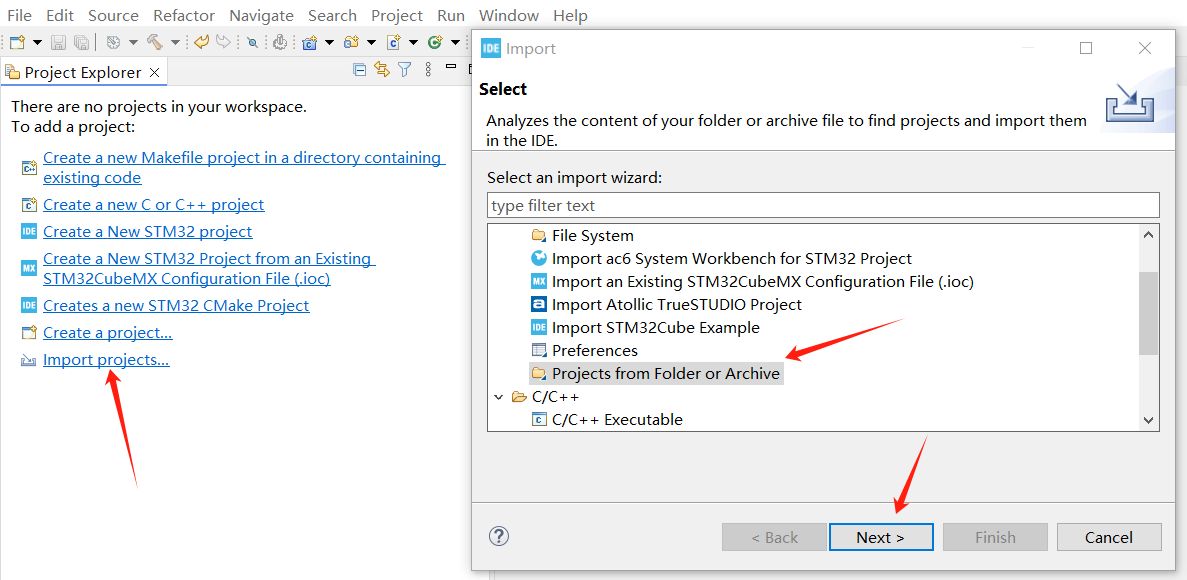
买的ST-Link下载器自带固件不够新，需要更新，先把CubeIDE更新到1.18.1然后再在软件里面点击Help/ST-Link更新即可，如图



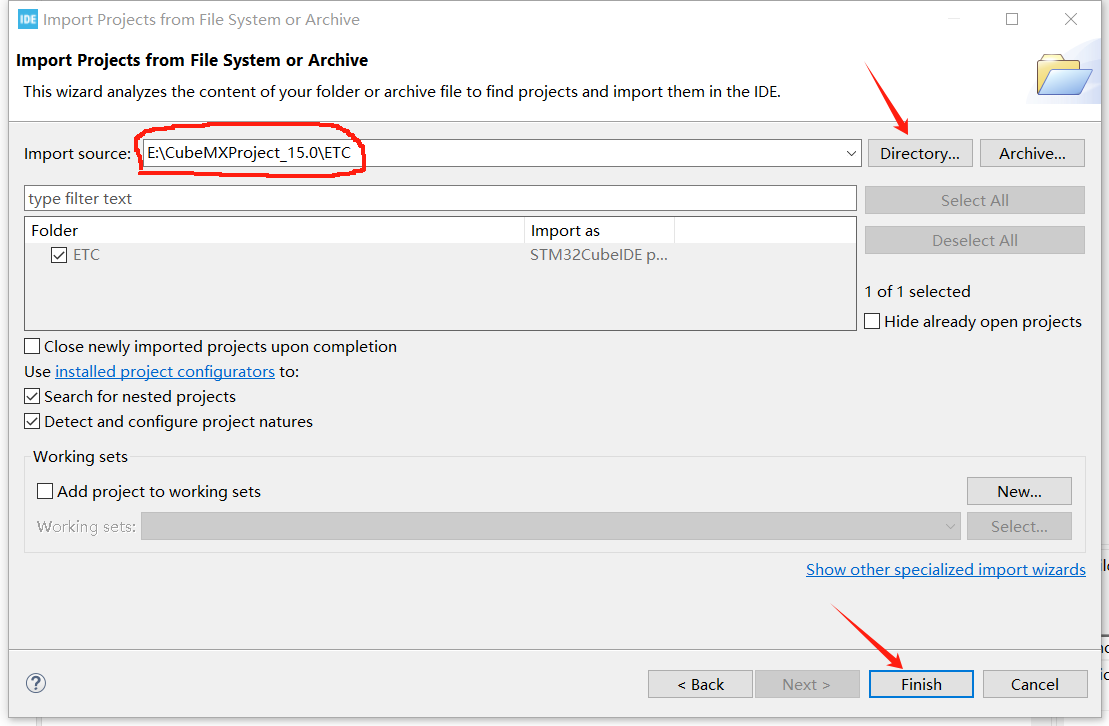


接着就是项目导入，导入流程如下：

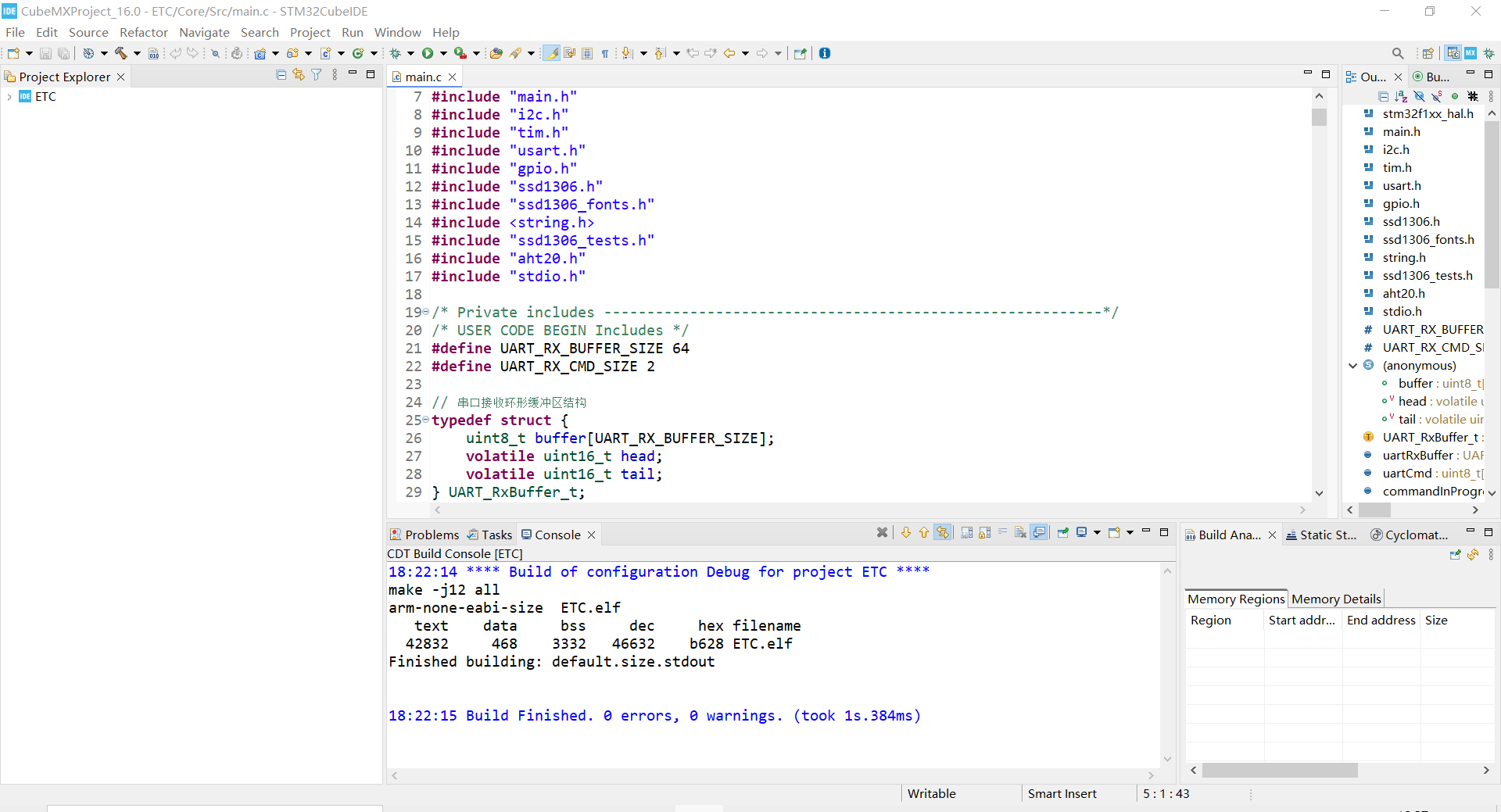
先点击import projects,然后点击projects from…之后Next



点击上图的Next之后点击Directory，然后选择你保存的项目也就是“ETC”(每个人保存的路径不一定一样，看你自己保存在哪的)，之后点击Finish



然后编译(点左上角那个小锤子)，下图就是编译成功的显示



之后点击下载即可（下载按钮和下图的绿色暂停按钮一样，在小锤子右边好几个身位），弹窗勾选Enable auto build然后点击OK即可



下图是下载成功的显示图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

最小系统板的代码下载完，还有个语音模块的数据需要处理，将JQ8900通过数据线连接到电脑上（该数据线必须是能够传输数据的数据线，而不是那种只能供电的普通线，普通线不会显示已插入u盘），然后将下面的文件全部复制到语音模块自带的内存卡中，直接替换原有的所有文件，配置好是下面这个样子，我们只用到了第2、3、4段语音。



至此32端的代码就已经欧克了，上电就能直接使用了。

不过里面的ETC.ioc这个图形界面配置不能直接套用，因为我当时只用它来初始化了最开始的引脚，后面就是直接在代码端复制修改了，所以有些引脚注释是不对的，直接用ioc文件生成的初始化代码也不对。

**关于树莓派端的代码使用，我准备了三种方式供不同人群使用：**

**第一种纯小白：**我直接提供了整个已经配置好的系统，可以直接下载烧入，链接如下

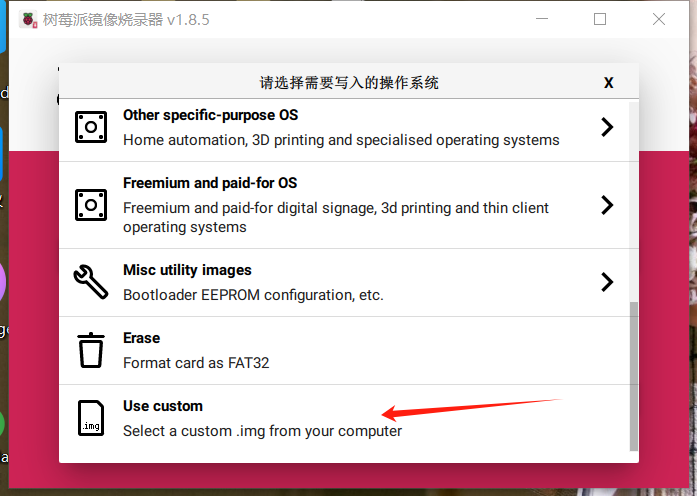
https://pan.baidu.com/s/1P1Ga9XHYgDXEcnvbeHr1dA?pwd=fct7

提取码：fct7

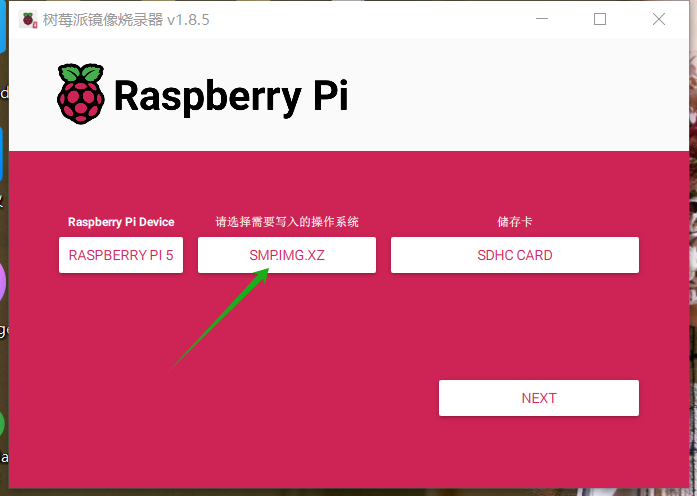
只需要在你自己电脑上使用树莓派专用烧写软件Raspberry Pi Imager即可完成（卖家会告诉你怎么用），以防万一还是说一下吧，打开该软件后一共有三个需要选择的，第一个选树莓派5，第二个选我给的SMP.IMG.XZ，第三个选你要烧到哪个位置，也就是你的SD卡（看大小也能看出来）

整体意思就是你要把适配树莓派5的SMP.IMG.XZ系统烧录到你选择的SD卡里面。

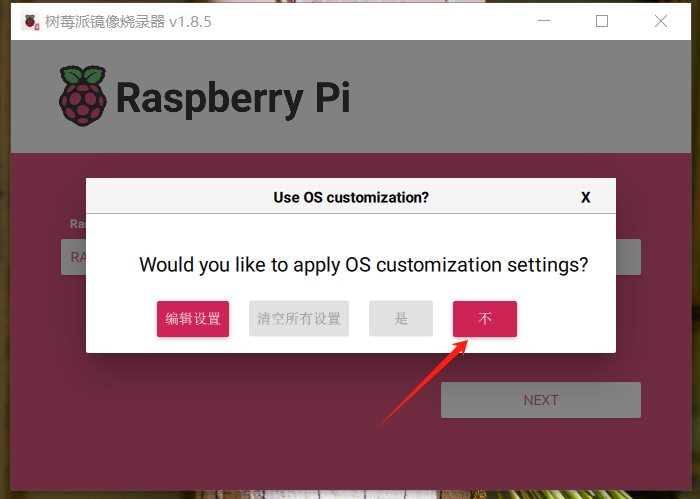
第一个太简单就没截图，下图是第二个，先点击Use custom再选择SMP.IMG.XZ（找到你当时放SMP.IMG.XZ的位置）



就比如下面这样，就是选择好了要烧录SMP.IMG.XZ这个系统，然后第三个选择SD卡，太简单没放图了。



选好之后点击NEXT，如果弹出下面这个对话框记得不要更改设置



然后等待即可，烧录好之后把SD卡插到树莓派上上电启动。

启动后还是需要可视化来方便操作的，要么使用VNC远程操控要么再接一个显示屏，如果使用VNC进行连接，账号是fct，密码是123456，记得操控别人的和被别人操控的得处于统一局域网下，中途有警告不用管，直接继续即可。

图形用户界面, 应用程序

描述已自动生成

进入系统后在终端输入命令运行代码(cd后面的命令不一定和我的一样，要看你具体把文件夹放在哪里的，放聪明一点不要完全不动脑子，要知道具体命令是干啥用的，好吧考虑到你是纯小白，cd后面的pi要换成fct，因为我给的系统就是这样放的)

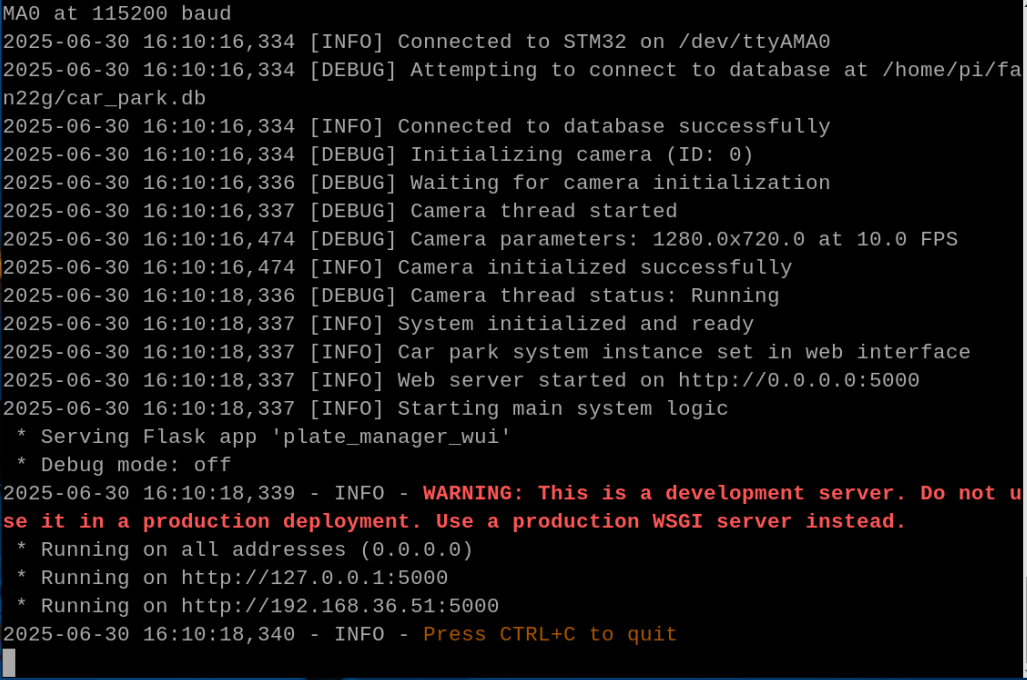
图形用户界面, 文本, 应用程序

描述已自动生成

首次运行代码会给你下载一些OCR的包，很快的（第一种方式没有这个，因为已经下好了）



当出现下面这个页面就代表成功了

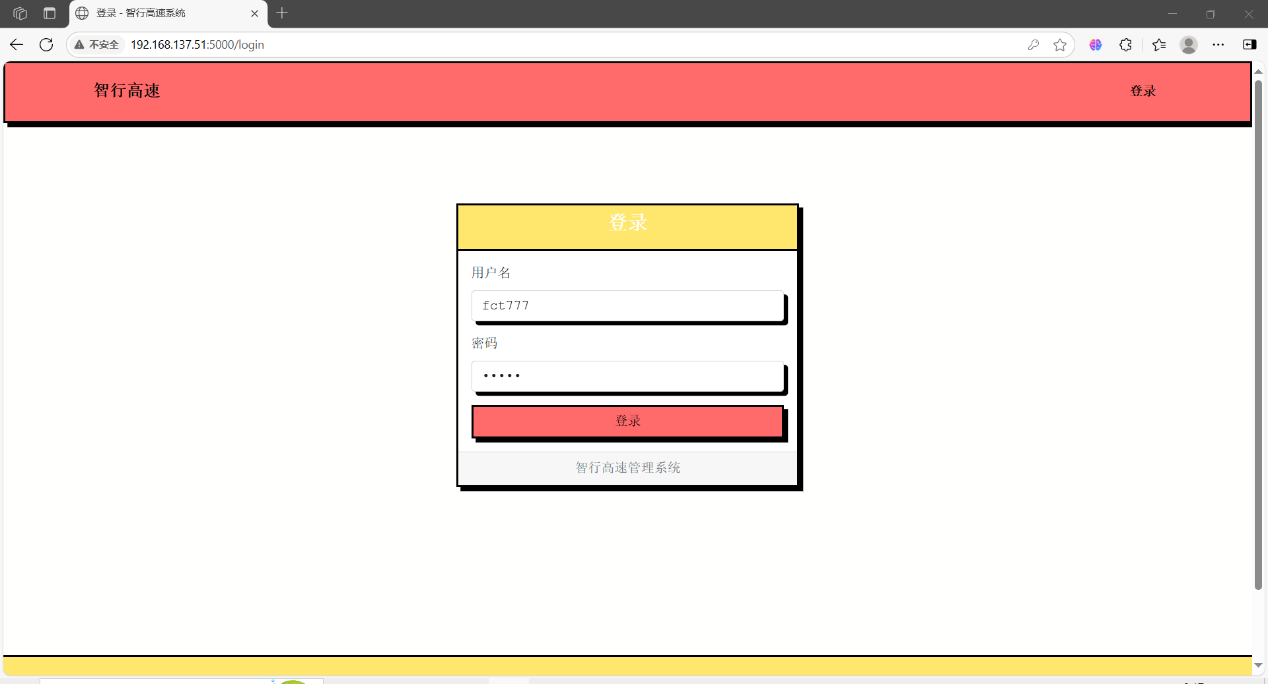


此时就可以使用另一个设备（与树莓派在同一局域网下）输入（树莓派的ip地址:5000）登录网页端进行查看了。

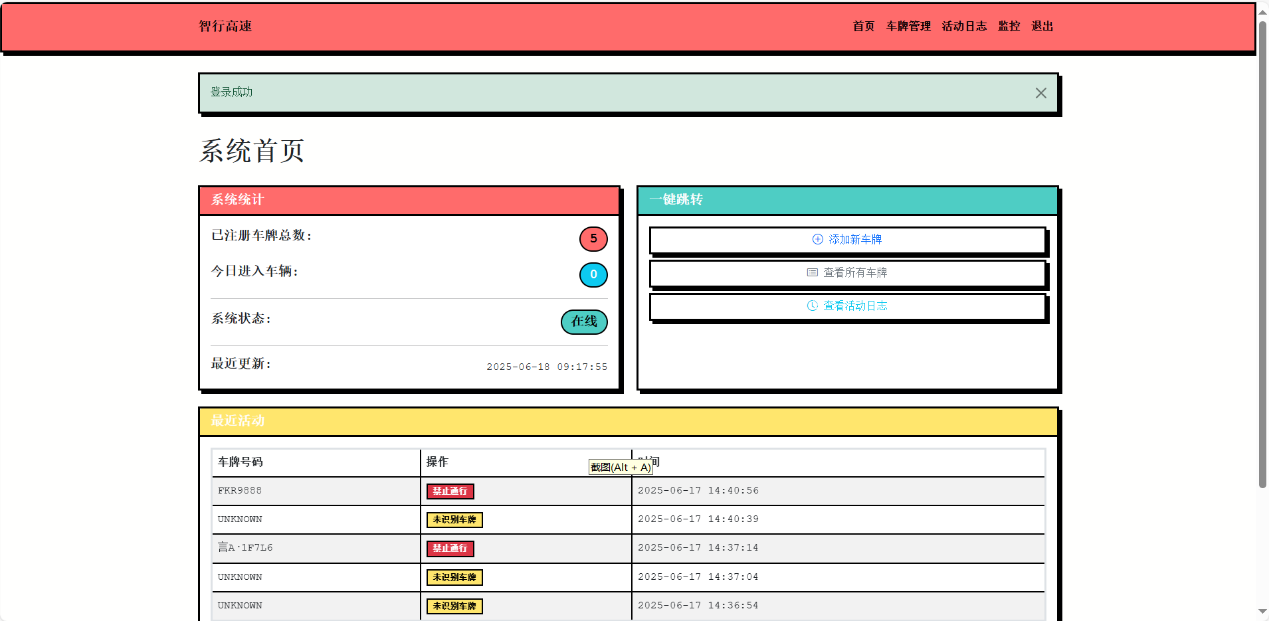
图里提示的127.0.0.1:5000是直接在树莓派查看网页端的网址，其他设备使用下面那个，根据自己树莓派的提示来输入。

然后网页端的登录如图所示（不要纠结为什么从36变成了137，这是换了一个ip地址的）

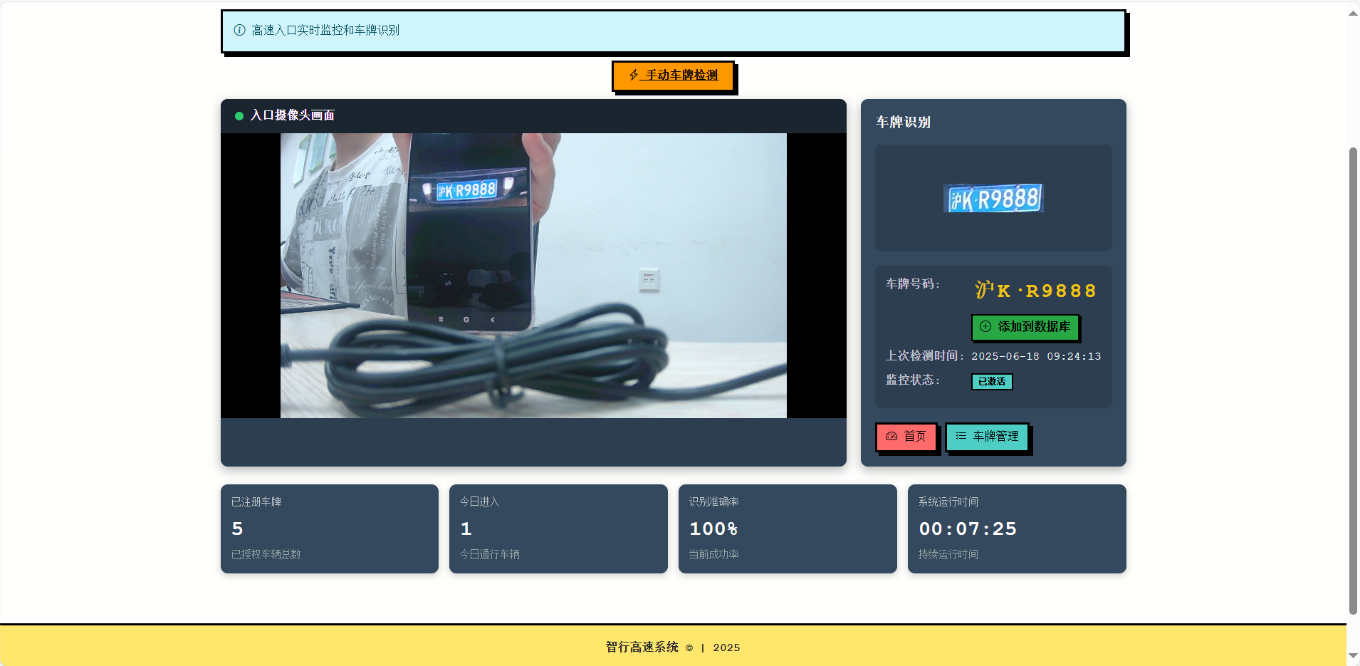
账号密码一样的，都是fct777，可以在plate\_manager\_wui.py文件里面修改。



登录之后就是系统首页



点击监控按钮就进入到了下面这个页面，包括实时显示、车牌信息识别结果、一键添加车牌以及手动车牌检测，还没连接下位机的话可以先试试手动检测，也能够执行检测线程，手动检测是刷新整个页面，所以响应速度较快，如果是自动检测，则受制于树莓派性能影响，图片出现得较慢，解决办法是给树莓派降温，多试几次，或者Ctrl+F5刷新一下网页，图片可有可无，主要是识别后的车牌信息才是关键，因为这个信息是文字，所以性能影响不大。



下面就是数据库，只有在里面的车牌才被允许通行



剩下的就是连接下位机了，二者相连就是使用三根线将树莓派与STM32端连接起来，具体就是GND连GND,RX连TX,TX连RX这样相连，板子上我只标注了RX/SMP，即最左边是RX，依次是RX/TX/GND。连接好之后最好在32端使用其自带的USB口进行供电，因为外设较多，下载器给的电不够，以上都做好之后你就玩吧。

**第二种方式是给有基础的人群准备的：**

默认你已经装好了系统，只需要安装miniconda，然后把我给的yolo文件夹放到…/miniconda/envs文件里，再然后把我给的fan22g文件放到主文件夹里（路径越短越好，后面运行时少敲命令），最后启用ttyAMA0即可。（因为我创的虚拟环境“yolo”文件夹里面包含符号链接，导致我上传的文件总是少一部分，可以在后面看到，慎用）

首先是安装miniconda

安装miniconda可以先在终端输入

uname -a

来查看自己的系统版本是不是aarch64，一般都是，不是就重新换一个系统；

然后是在终端输入以下命令

wget <https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-aarch64.sh>

这行命令是直接从官网下载最新的miniconda

我当时下的是2025-06-24 13:32:08这个版本

哈希值是368d5250682c12d3e1810e8cf5ce4f08e95ccc56c96dcf559567a6b4f72b203d

之后运行下面的命令直接安装，

bash ./Miniconda3-latest-Linux-aarch64.sh

之后一直按回车确认，当左下角的More变成END的时候，可以按q键退出，之后询问的就都输入yes回车即可。

当出现Thank you for installing Miniconda3!字样的时候，证明我们的Miniconda已经安装好了，重启终端即可，就会进入base环境

之后是yolo文件的复制，不过需要注意一点，这个yolo文件里面包含符号链接，如果你的u盘是FAT32的文件格式，会缺少一部分文件，如下图，多的那个才是正确的（72935），所以需要你下载之后仔细对比一下文件数量（我这是在windows系统对比的数量，你们也可以确认一下）。

yolo文件上传会缺失一部分文件，所以我并没有给yolo文件的下载链接，实在要用该方法，我建议准备两张SD卡，一张专门烧录第一种方式我给的系统，然后从里面将/miniconda3/envs路径下的yolo文件夹拷出来即可，记得使用NTFS格式的U盘或硬盘（下面就能跟着做了，如果后续能解决我会上传）

图形用户界面, 应用程序

描述已自动生成

如果不对（只能多，不能少），要么换其他文件格式的硬盘重新下载，要么想想其他替代办法，比如直接无线传输（SSH），实测缺了这几个文件就会出现问题，我还测试了exFAT格式的硬盘依旧不行，压缩之后也不行，期待你们的解法，我使用的NTFS格式的硬盘就没问题，就是直接将~~我给的这个~~yolo文件下载到NTFS格式的硬盘里面，再将该硬盘插到树莓派，然后复制粘贴，当然复制之前也可以检查一下文件数目。

确定数目对的上就把yolo文件复制到…/miniconda/envs文件里

图形用户界面, 应用程序

描述已自动生成

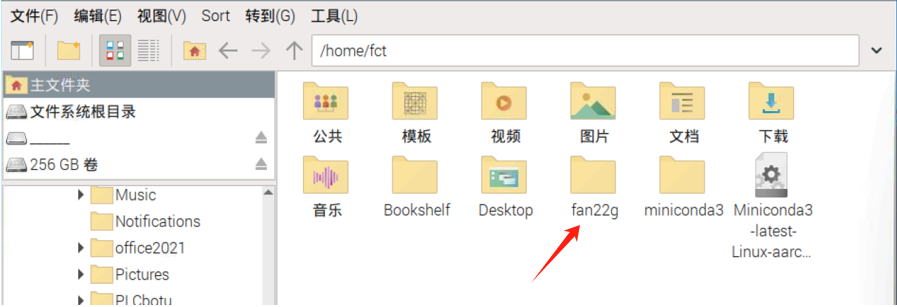
这就相当于你创建了一个名为”yolo“的虚拟环境，里面代码所需的各种包和库都配好了的，完成后就可以直接在终端输入以下命令；

conda activate yolo

进入该虚拟环境了

然后将fan22g文件复制到主文件夹里（比如/home/fct）如图

作用跟在CubeIDE导入32的项目代码一样



之后是启用ttyAMA0。（怎么启用搜一下，同时记得将当前用户加入dialout组，这样就不用每次都访问权限了，不放图的原因是AI说得比我好太多了）

以上都完成就是直接连线运行代码了，如下图（剩下的就参考第一种方式操作即可）

图形用户界面, 文本, 应用程序

描述已自动生成

**第三种是针对想走我来时路的道友：**

首先是烧系统，装上系统后需要安装miniconda，然后在miniconda里面创建一个虚拟环境，比如命名为yolo，然后再在yolo虚拟环境里面安装Ultralytics软件包来允许使用训练好的yolo模型；之后还需要在yolo虚拟环境里面下载我们代码所使用的各种库和依赖项，然后是启用ttyAMA0，最后是放入fan22g文件到你喜欢的位置。

首先安装miniconda可以先在终端输入

uname -a

来查看自己的系统版本是不是aarch64，一般都是，不是就重新换一个系统；

然后是在终端输入以下命令

wget <https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-aarch64.sh>

这行命令是直接从官网下载最新的miniconda

我当时下的是2025-06-24 13:32:08这个版本

哈希值是368d5250682c12d3e1810e8cf5ce4f08e95ccc56c96dcf559567a6b4f72b203d

之后运行下面的命令直接安装，

bash ./Miniconda3-latest-Linux-aarch64.sh

之后一直按回车确认，当左下角的More变成END的时候，可以按q键退出，然后继续下一步，之后询问的就都输入yes回车即可。

当出现Thank you for installing Miniconda3!字样的时候，证明我们的Miniconda已经安装好了。此时更新一下bashrc文件，输入下面的命令

source ~/.bashrc

运行之后整个的miniconda就成功安装了，建议重启后再输入以下命令

conda create -n yolo python=3.12.11

我的就是这个版本的python，这就创建了一个名为“yolo”的虚拟环境，之后就是输入以下命令进入该虚拟环境

conda activate yolo

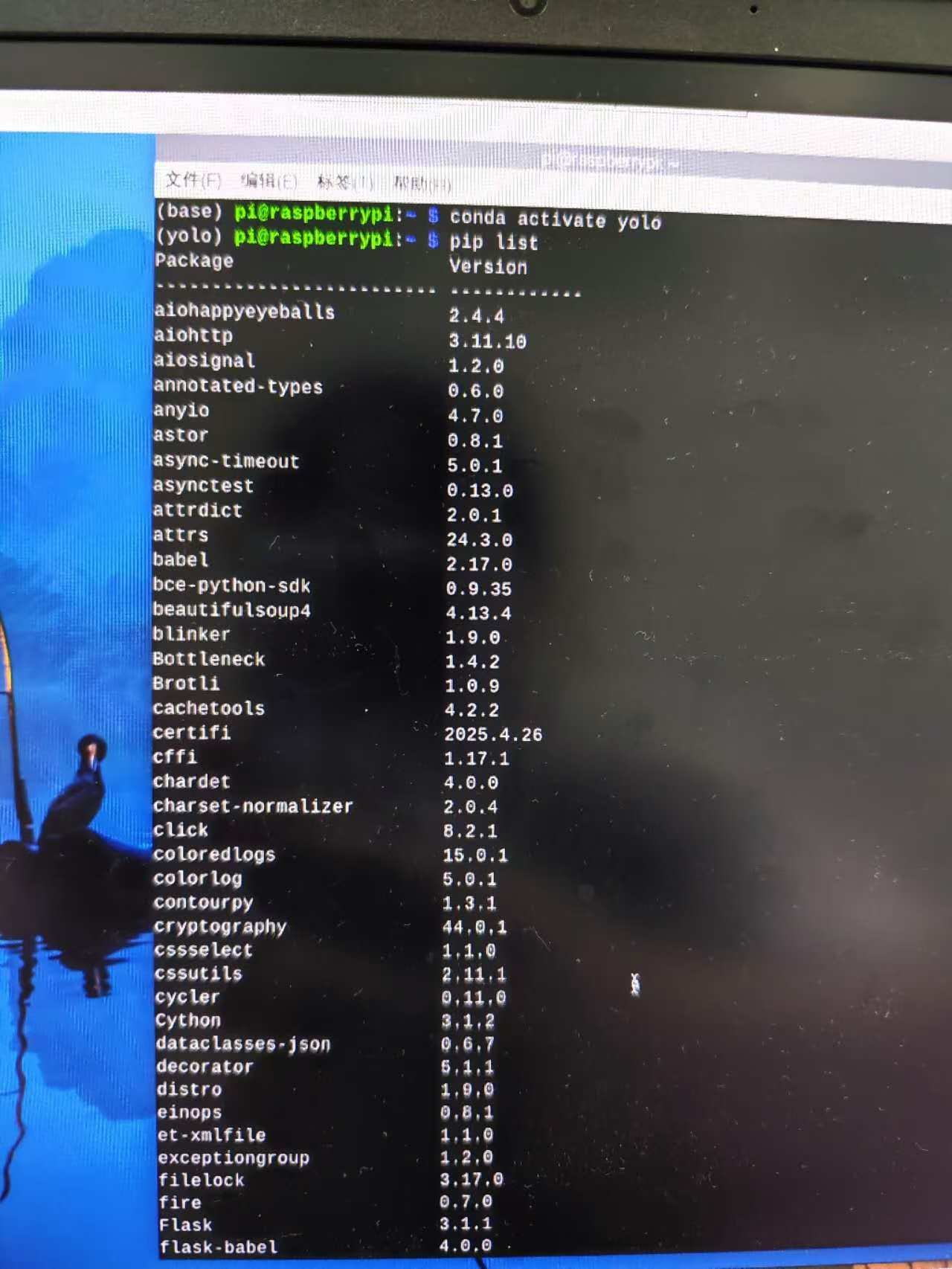
进入后就可以安装Ultralytics 软件包了

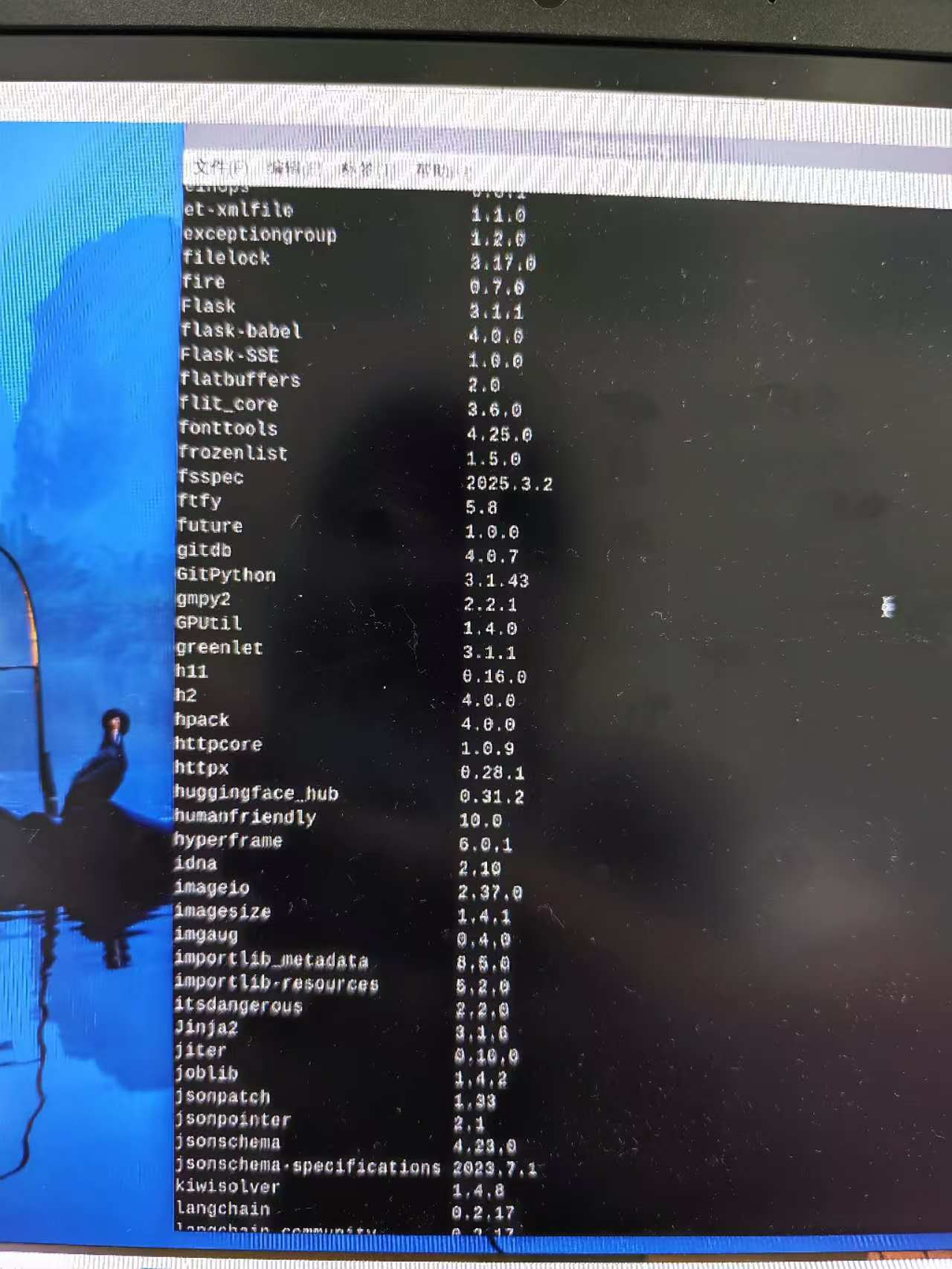
其中具体操作可以按照yolo官网里面的教程安装，如图所示，其中的[export]不要输入

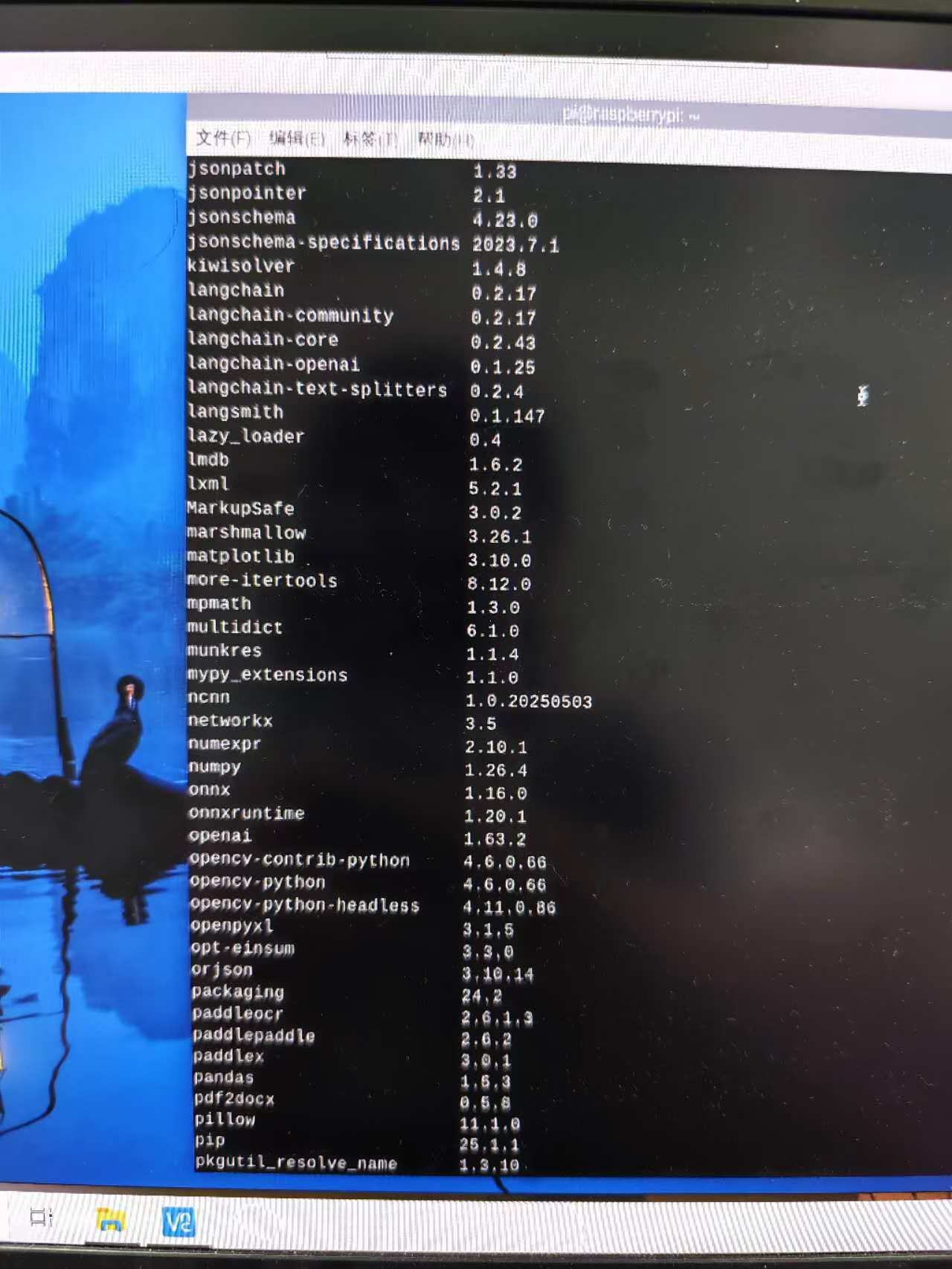
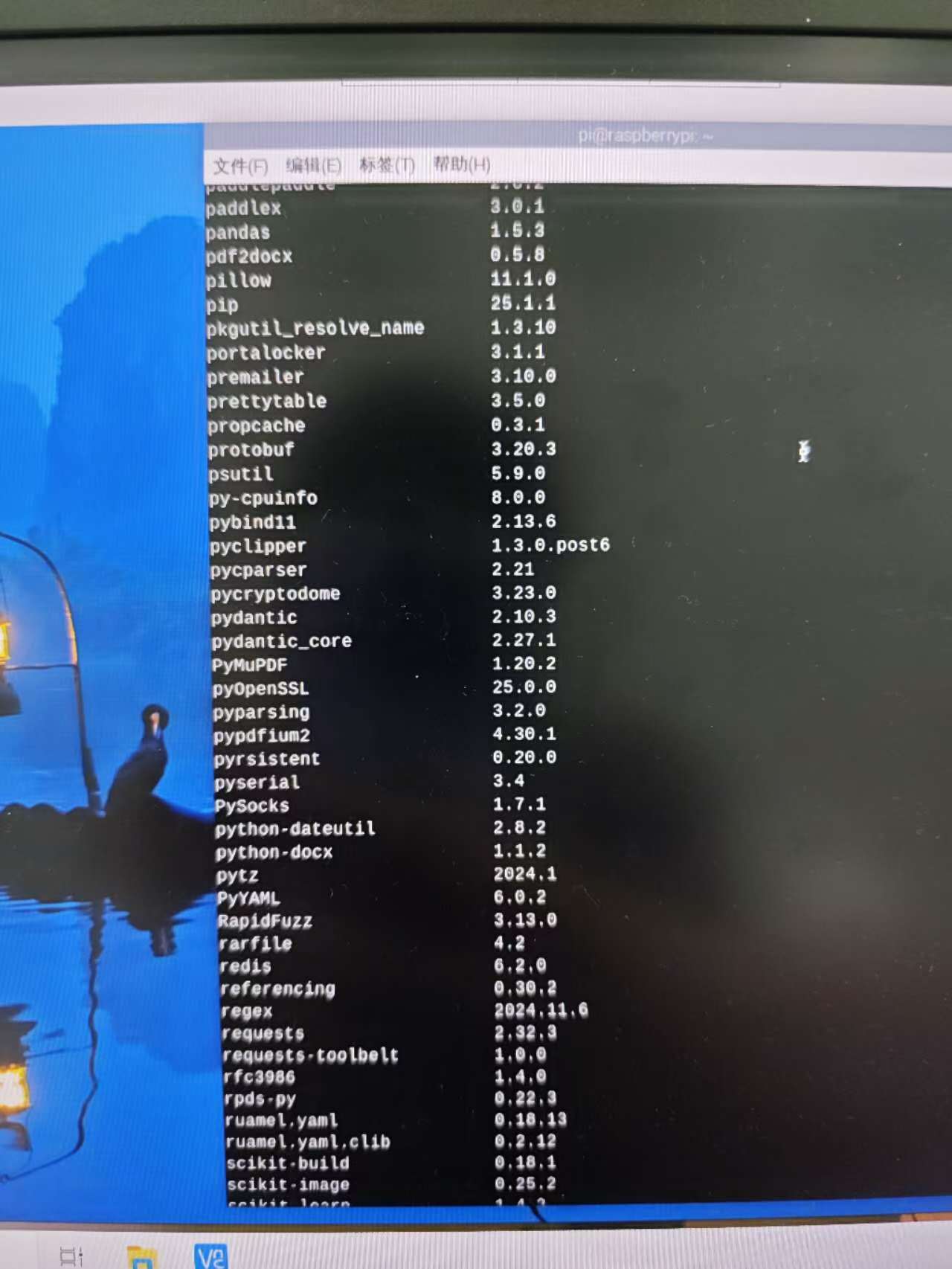


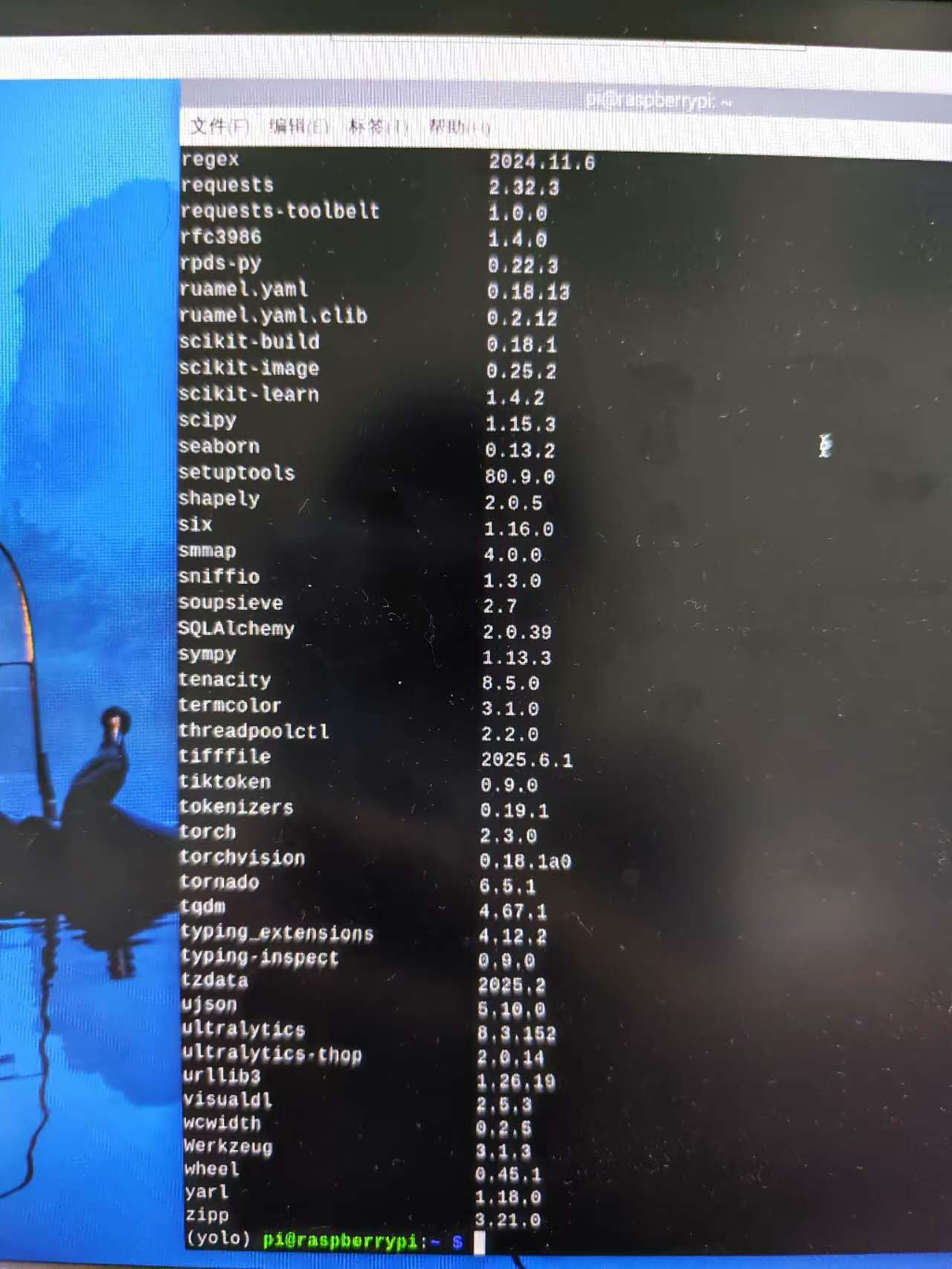
把上面的下载完成后再重启，就完成了Ultralytics 软件包的安装了，后面就是安装代码所需的库和依赖项了。

关于库和依赖项不用完全按照我下面的图片的库来下载，因为有些其实并没有派上用场，可以看看代码的最开头，导入了哪些包，我的建议是直接运行代码，然后提示你缺啥你再下啥，至于下的版本可以参考下面的图片，版本最好是和我的一样，尤其是paddle和paddleocr的版本（这两个我搞了两天才找到可用的版本组合），如图，不要看到图里面比较多就畏缩，都走到这一步了就很快了，里面很多都是依赖项，你下载一个主的，其他的都是附加一起下载的，也就二十个左右是需要你下载的，很快的。

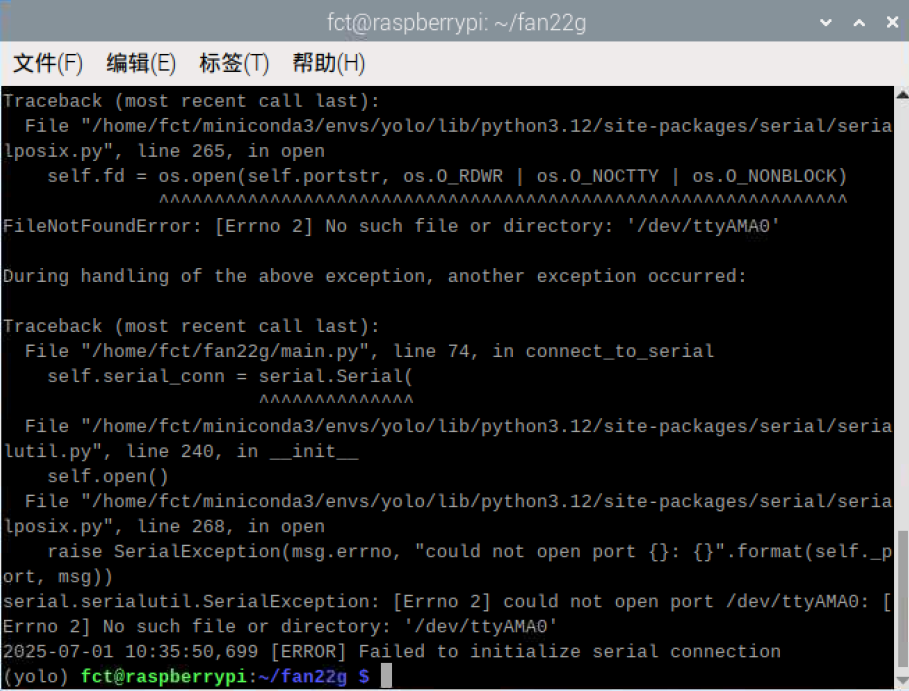








之后就是得修改一下串口配置，下面就是没有启用ttyAMA0会出现的报错，如果是自己装的系统，就得启用一下（怎么启用搜一下即可），记得将当前用户加入dialout组。



解决之后就是把我的fan22g文件放到你喜欢的位置了，然后就完事了，剩下的就是连线、运行看效果了（可以参考第一种方式）。

走到这，恭喜你，世另我~

**下面与项目无关，纯属个人建议**

致未来的你：

此时本人正是一个24岁的青年（风华正茂），而我正在做这件事的时候你可能正在上研究生，读大学甚至读高中，不过没关系，此刻你终究还是走到了我的面前，我知道你肯定也走了和我相似的路，看到了我曾经欣赏过的风景，所以才有我们的相遇。

我这一路走来，从最开始的三五成群，到最后的独自前行，有人中途离场，也有人半路加入，有主观原因，也有客观因素，也许你看到这的时候我早已毕业、走进工作、娶妻生子甚至不再碰嵌入式，不过我不会忘记这次经历，哪怕现在只有我一个人记得，因为我相信，未来一定还有你替我继续记忆这段经历。现在我倒是能够理解古人为什么对流芳百世、名垂青史这么执着了，因为一个人，别说在宇宙中了，哪怕是在地球上也是如此的渺小，生命周期如此的短暂，所以只能做一些事来让别人记住自己，代表自己曾经来过，不然，当世界上最后一个记得你的人死去，那你就真的消失了，以后再也不会有人提起你了。

人们都说，年轻的时候要多去做一些有意思的事，这样老了以后才有值得回味的东西。从6月7号到如今的7月4号，这段时间我全身心地投入到这个项目，我可以说，这是我这学期做的最有意义的一件事。而我想，这件事情一定会在我以后的回忆中熠熠生辉，甚至随着时间的推移在我的记忆中越来越牛逼。你想想，假如三年五年后的你复现了我这个项目，让“智行高速”重现人间，就相当于两个原本没有任何交集的人类进行了一场跨越时间和空间的交流，哪怕我们相隔千里，哪怕我们正处于不同的人生轨道，但这一刻，我们两人的轨道，重合了，这是多么浪漫的一件事！

那么，我们来玩个游戏让这件事更浪漫一些，我在下面放了几个二维码，分别是三毛钱的，两毛钱的，一毛钱的，还有五分钱的，三分钱的，你如果是使用第一种方式完成的，就扫三毛钱的（小白就多交点学费），第二种方式完成的就扫两毛钱的，第三种方式完成的道友就扫一毛钱的，要是只完成了树莓派端的就扫五分钱的，只完成了32端的就扫三分钱的。

当然当然，你要是不玩也没关系，主要目的是为了统计，我很想看看有多少人能走到我面前。

好吧，其实我还是有私心的，就比如以后我正在为论文绞尽脑汁时，为柴米油盐奔波劳累时，在家庭琐事的漩涡中挣扎时，在辅导孩子功课崩溃绝望时，突然，一则重磅消息袭来，某年某月某日，收款到账一毛钱。这意味着，我的“智行高速”在世界某个角落重生了，找到新的主人了，那时我的思绪就会被强制拉回到这段经历的回忆中，忆往昔，峥嵘岁月，越想越会觉得自己牛逼，这就像又一个年轻时候风华正茂的自己回来了，那么还有什么坎过不去呢。（所以私心就是现在的我在建议未来的你们给以后的我打气）

第一种方式复现 第二种方式复现



第三种方式复现 只复现了树莓派端



只复现了32端



（PS：每一笔进账我都会截图并且拿小本本记下来，当然，一定是满足上面的条件你才扫码哈，别看两眼觉得好玩你就扫了，要是觉得有帮助点个星星就好，让其他人也参与进来）

同时我也希望有同道帮我继续完善这个项目，因为我还想被人记住得久一些，道友你也想的吧，项目越好才会有更多人知道并记住我们，我们才能“活”得更久一些。我提供几个思路：首先是环境里面的Paddle和PaddleOCR的包是比较老比较难下的，最容易出问题的也就是这里，所以可以从代码层面改进，将里面老旧的参数，函数可以替换成最新的，方便后人复现；还有就是可视化使用的Flask也是不被长期支持的，保不准哪天就不能用了，这也可以换成更稳定更新的；还有就是功能的增加，现在是只回传两种信号（OK,NO），对应两种结果，但是还有第三种结果，“未识别车牌”，也应该单独处理，我已经在语音模块预留了他的接口，方便后期拓展；其他的就你们自由发挥了，其实都不难，都只是一些基础的运用，我只是实在没有精力去完善它了，下学期还要忙着找工作和中期呢，我自己正在研究的基于深度强化学习的动态避障也需要完善（这个等我毕业了也会开源出来，不难，只能帮你们省“亿”丢丢时间罢了）。如果有人完善出来，我会联系你并且分你一半已收到的钱款，这都是荣誉啊兄台。

最后我还是得说嵌入式真的能让人学到很多东西，逻辑、耐心、恒心、思维等等，都能在这里得到培养，尤其是排查错误的能力哈哈哈，因为你生活中肯定会遇到各种错误，而解决错误的前提就是找到错误。

最最最后，干杯！这一杯叫共勉~