00A0 2203∃ 2200∀ 2286⊆ 2713x 27FA⟺ 221A$\surd$ 221B$\sqrt[3]{}$ 2295⊕ 2297⊗ UTF8gbsn

SunFounder 帕克智能小车

www.sunfounder.com

2024 年 03 月 05 日

Contents

1 介绍 **3**
- 1.1 自动驾驶汽车的历史 . . . 3
- 1.2 关于帕克 . . . 4
- 1.3 深度学习和神经网络 . . . 5

2 部件清单和装配说明 **7**

3 关于 Robot HAT **9**

4 玩转 EzBlock **13**
- 4.1 EzBlock 快速指南 . . . 13
- 4.2 下载 EzBlock Studio . . . 21
- 4.3 快速配置 . . . 21
- 4.4 打开或新建项目 . . . 25
- 4.5 移动 . . . 28
- 4.6 远程遥控 . . . 32
- 4.7 测试超声波模块 . . . 34
- 4.8 测试灰度模块 . . . 36
- 4.9 颜色检测 . . . 37
- 4.10 人脸检测 . . . 41
- 4.11 音效 . . . 44
- 4.12 背景音乐 . . . 46
- 4.13 问好 . . . 47
- 4.14 音乐车 . . . 49
- 4.15 悬崖检测 . . . 51
- 4.16 矿车 . . . 53
- 4.17 矿车 + . . . 57
- 4.18 斗牛 . . . 62
- 4.19 礼让行人 . . . 64
- 4.20 交通标志检测 . . . 66
- 4.21 定向越野 . . . 71

5 玩转 Python **77**
- 5.1 Python 快速指南 . . . 77
- 5.2 校准 PiCar-X . . . 111
- 5.3 让帕克动起来 . . . 115
- 5.4 避障 . . . 117

5.5 线路跟踪 . . . 118
5.6 文字转语音 . . . 120
5.7 计算机视觉 . . . 121
5.8 颜色检测 . . . 124
5.9 人脸检测 . . . 128
5.10 视频车 . . . 131

6 附录 **135**
6.1 网页版 EzBlock . . . 135
6.2 apt 和 pip 更换国内源 . . . 139
6.3 Filezilla 软件 . . . 141
6.4 关于电池 . . . 143
6.5 PuTTY . . . 144
6.6 通过 Powershell 安装 OpenSSH . . . 146

7 疑难解答 **149**
7.1 当使用 VNC 时，我被提示桌面暂时无法显示? . . . 149
7.2 安装 EzBlock 操作系统后，舵机不能转到 0°? . . . 149
7.3 如何在 EzBlock 重新校准机器人 . . . 150
7.4 EzBlock 无法连接蓝牙? . . . 151
7.5 APP 搜索到蓝牙，但无法连接 . . . 152
7.6 配置 WIFI 后 APP 无法连接 . . . 152
7.7 为什么伺服机有时会无缘无故地返回到中间位置? . . . 152

8 版权声明 **153**

感谢你选择我们的帕克智能小车。

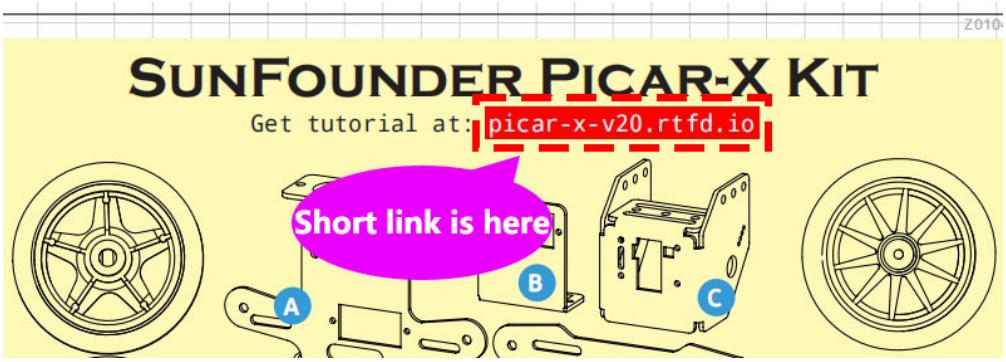
备注： 本文档支持以下语言版本。

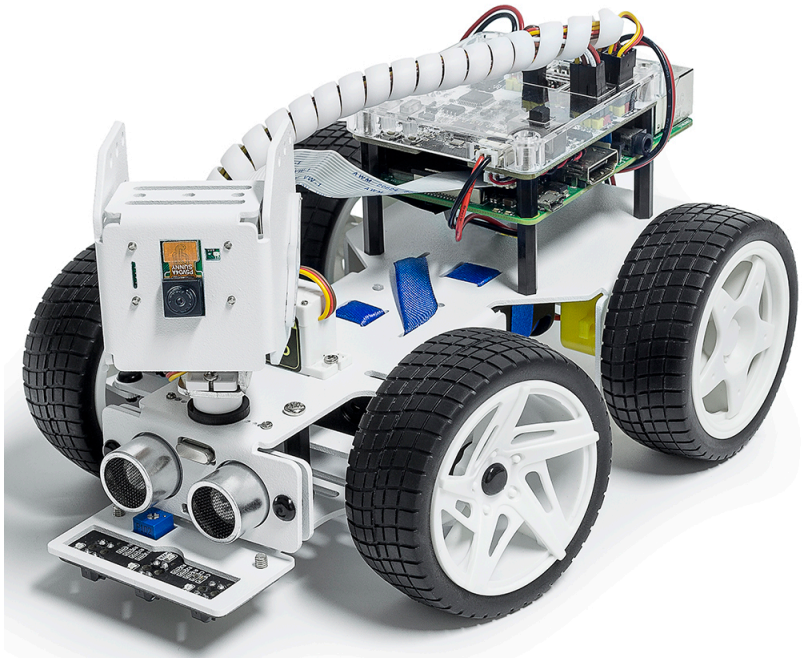
-
-
-

请点击相应的链接以选择您偏好的语言版本查阅。

警告： 我们提供两个版本的 PiCar-X。需要特别注意的是，每个版本在线教程中的脚本不能互换使用。为确保正确设置，您需要根据说明书中提供的短链接确定您的版本：

- 若链接为”picar-x.rtfd.io”，请继续阅读此教程。
- 若链接显示为”picar-x-v20.rtfd.io”，请前往查看教程。





帕克是树莓派平台的人工智能驱动的自动驾驶机器人小车。它拥有两轴相机模块、超声波模块和线路跟踪模

块可以提供颜色/脸部/交通标志的检测功能以及自动避开障碍物、自动跟踪线路等功能。

本教程包括组件清单、装配图、Robot HAT 介绍，以及基于图形化编程和 Python 语言的帕克编程教程。

编程部分分为两章：*玩转 EzBlock* 和*玩转 Python*。每个编程章节都允许用户从让帕克执行基本动作开始，一直到设计多人游戏，甚至更多的游戏!

EzBlock Studio 是 SunFounder 创建的一个软件开发平台，用于帮助初学者和新手开始在树莓派平台上进行编程。Ezbock Studio 有两种编程语言环境：一种是图形用户界面环境，一种是命令行 Python 环境。

EzBlock 几乎可用于所有类型的设备，包括 Mac、PC 和 Android。EzBlock 最近已经升级到 3.0 版本，它使用 WebSocket 提供蓝牙和 Wi-Fi 支持。

对于只想学习 Python 3.0 编程的用户，还包括树莓派操作系统的基础知识介绍，以及提供基本 Python 编程技能的教程。

目录

CHAPTER 1

介绍

1.1 自动驾驶汽车的历史

至少从 20 世纪 20 年代开始，就已经进行了自动驾驶汽车的实验。1950 年代进行了有希望的试验，此后工作一直在向前推进。第一辆自给自足和真正的自动驾驶汽车出现在 20 世纪 80 年代。1984 年，卡耐基梅隆大学的 Navlab 和 ALV 项目。以及 1987 年梅赛德斯-奔驰和慕尼黑联邦国防大学的 Eureka Prometheus 项目。自 20 世纪 80 年代末以来。许多研究机构和主要的汽车制造商已经开发出可以使用的自动驾驶车辆。包括。梅赛德斯-奔驰、通用汽车、大陆汽车系统、Autoliv 公司、博世、日产、丰田。奥迪、沃尔沃、帕尔马大学的 Vislab、牛津大学和谷歌。2013 年 7 月，Vislab 展示了 BRAiVE，一辆在向公众开放的混合交通路线上自主行驶的车辆。截至 2019 年，美国有 29 个州已经通过法律，允许自动驾驶汽车在公共道路上行驶。

一些联合国欧洲经济委员会成员和欧盟成员，包括英国。已经颁布了与自动驾驶和全自动驾驶汽车有关的规则和条例。在欧洲，比利时、法国、意大利和英国的城市已经制定了运营无人驾驶汽车运输系统的计划。而德国、荷兰和西班牙已经允许在公共交通中测试机器人汽车。2020 年，英国、欧盟和日本已经走上了规范自动驾驶汽车的轨道。

今天，自动驾驶汽车是目前最接近的技术革命。一些专家预测，到 2025 年，4 级汽车可能会进入市场。4 级汽车将允许司机将注意力完全转移到其他方面，只要系统运行正常，就不需要注意交通状况。

然而，尽管每天都收集了大量的数据，包括来自真实驾驶记录和模拟场景的训练数据，但自动驾驶汽车的人工智能模型的复杂性还没有完全满足。

根据兰德公司的报告，达到适当的自主学习水平需要来自数亿，甚至数千亿英里的训练数据，以建立一个可靠性的水平。

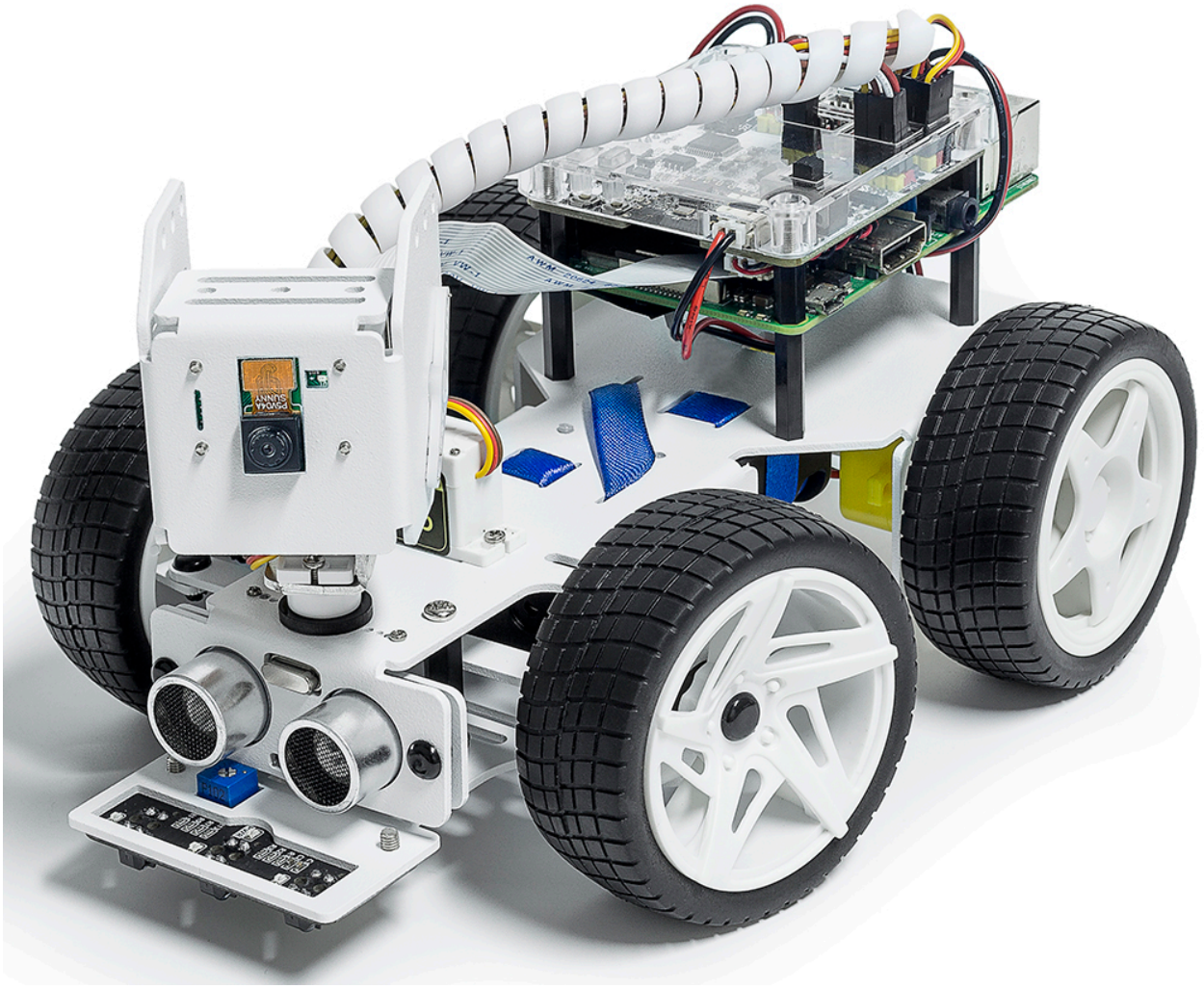
..‘兰德公司的报告 <https://www.rand.org/pubs/research_reports/RR1478.html>‘_

因此，尽管自动驾驶汽车的未来很有希望，令人振奋，但在技术成熟到可以完全进入自动驾驶汽车市场之前，还有很多年的发展时间要走。

让一项新兴技术迅速成熟的行之有效的方法是，通过最大限度地减少市场准入要求，使每个人都能轻松获得该技术。这就是 SunFounder 推出帕克的动机。

SunFounder 的目标是帮助初学者、新手和那些只是想了解自动驾驶的人，了解自动驾驶汽车的开发过程、技术和最新创新。

1.2 关于帕克



帕克是树莓派平台的人工智能控制的自动驾驶机器人汽车，在此基础上，树莓派充当控制中心。帕克的 2 轴摄像头模块、超声波模块和线路跟踪模块可以提供颜色/脸部/交通标志检测、自动避障、自动线路跟踪等功能。

通过 SunFounder 设计的 Robot HAT 板，帕克集成了左/右驱动电机、用于转向的伺服电机和摄像头的平移/倾斜功能，并预先设置了 Robot HAT 的 ADC、PWM 和数字 I2C 引脚，以实现对 Raspberry Pi 标准功能的扩展。一个扬声器和一个蓝牙芯片都被设计到 Robot HAT 中，用于远程控制文字转语音、声音效果，甚至背景音乐功能。

帕克的所有功能，包括 GPIO 控制、计算机视觉和深度学习，都是通过开源的 Python 编程语言、OpenCV 的计算机视觉库软件和谷歌的 TensorFlow 深度学习框架实现的。其他软件也被包括在内，以优化帕克的能力，让用户有一个接近无限的学习环境。

1.3 深度学习和神经网络

要了解更多关于深度学习和神经网络的信息，SunFounder 推荐以下资源。

机器学习--Andrew Ng： 该课程对机器学习、数据挖掘和统计模式识别进行了广泛介绍。

神经网络和深度学习：这本电子书涵盖了神经网络和深度学习，前者是一种受生物启发的编程范式，使计算机能够从观察数据中学习，后者是神经网络中机器学习的一套强大技术。

重新思考计算机视觉的初始架构： 这篇高水平的白皮书探讨了用户可以通过因子化卷积和积极的正则化，尽可能有效地利用增加的计算来扩展网络的方法。

CHAPTER 2

部件清单和装配说明

在组装帕克之前，请首先确认所有的部件和组件都已包括在内。如果有任何缺失或损坏的部件，请立即与售后服务联系，以尽快解决这个问题。

请按照以下 PDF 文件上的步骤进行组装说明：

- [PDF] 帕克组装教程。

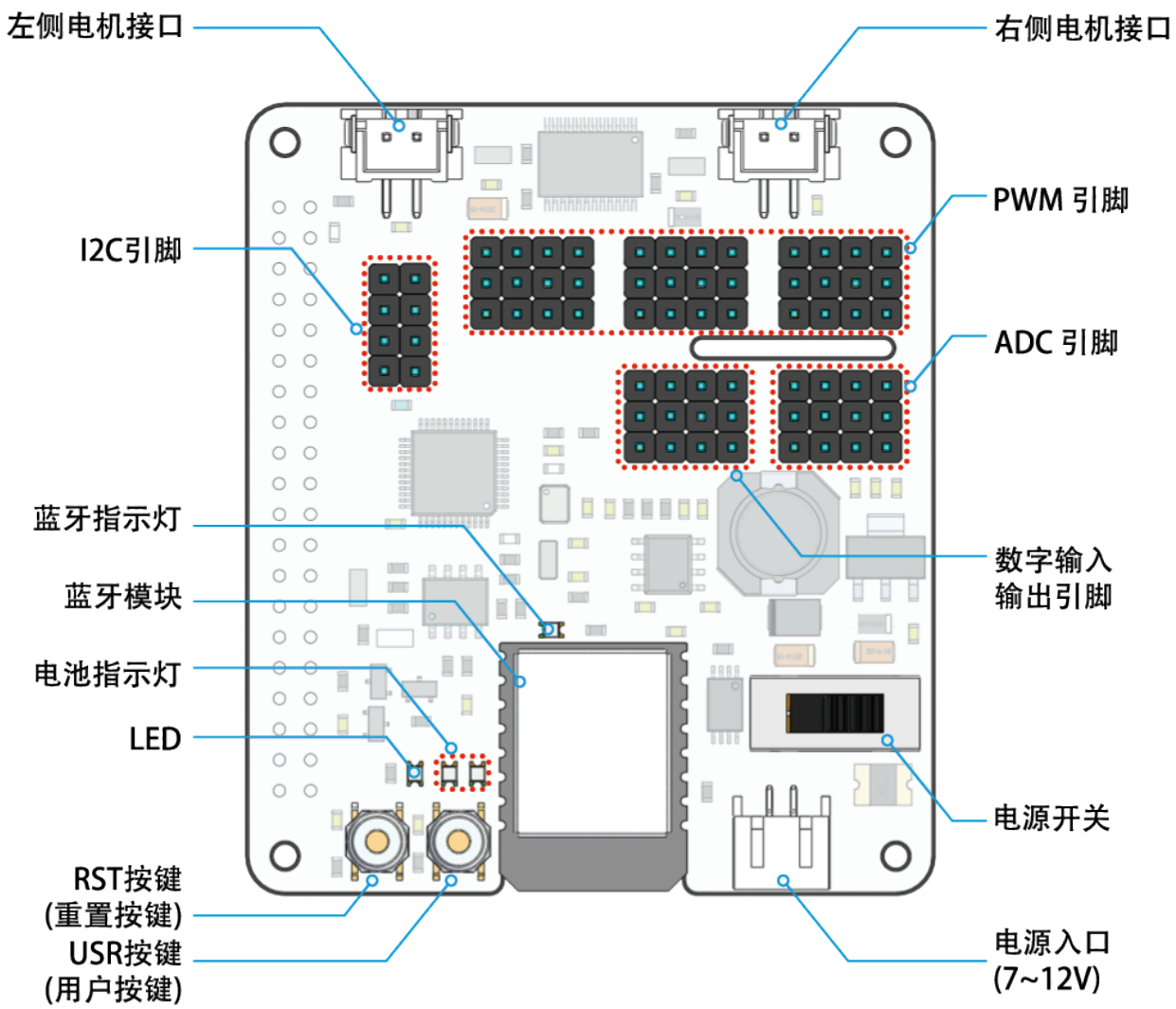
备注： 如果舵机在组装后已经通过 Robot HAT 通电，请不要手动强行打开转向器，因为这可能会导致舵机损坏。

备注：

1. 在组装之前，你需要购买 2 个 18650 电池并充分充电，参考关于电池。
2. Robot HAT 不能给电池充电，所以你需要同时购买一个电池充电器。

CHAPTER 3

关于 Robot HAT



RST 按钮

- 短按 RST 按钮将导致所有正在运行的程序重置。
- 长按 RST 按钮直到 LED 亮起，然后松开将断开 Robot HAT 的蓝牙芯片。

USR 按钮

- USR 按钮的功能可通过编程进行配置。（按下导致输入 `0` ，松开产生 `1` 输入）

LED

- 通过编程配置（输出 `1` 打开 LED，输出 `0` 关闭 LED。）

电池指示灯

- 电池电压高于 7.8V 将点亮两个 LED 指示灯。电池电压在 6.7V 到 7.8V 之间只会点亮一个 LED，低于 6.7V 的电压将关闭两个 LED。

蓝牙指示灯

- 蓝牙指示灯 LED 将在蓝牙连接稳定时保持亮起，并在信号传输期间快速闪烁。如果蓝牙断开连接，LED 将每隔 1 秒闪烁一次。

CHAPTER 4

玩转 EzBlock

EzBlock 是 SunFounder 所使用的一个软件开发平台，用于帮助初学者和新手在树莓派平台上开始学习编程。Ezbock 有两种编程语言环境：一种是图形用户界面环境，一种是命令行 Python 环境。

EzBlock 几乎可用于所有类型的设备，包括 Mac、PC 和 Android。

在使用 EzBlock 之前，请先按照以下教程来完成 EzBlock 下载，安装和使用。

4.1 EzBlock 快速指南

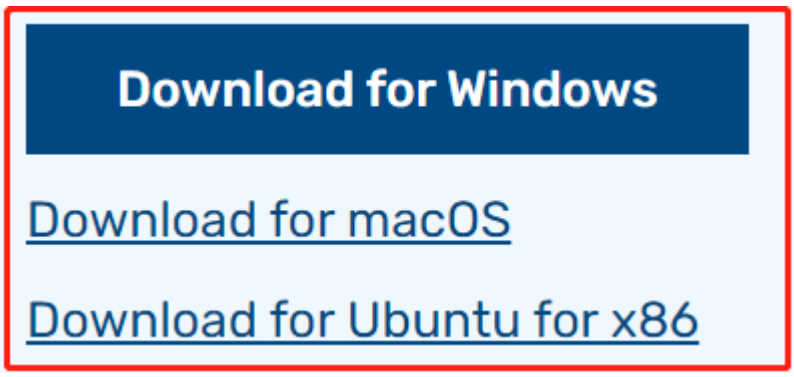
这里是使用 EzBlock 的一些使用指南。在组装的时候，你只需要完成*安装 EzBlock 镜像* 和*舵机调零*。

建议按顺序阅读课程内容。

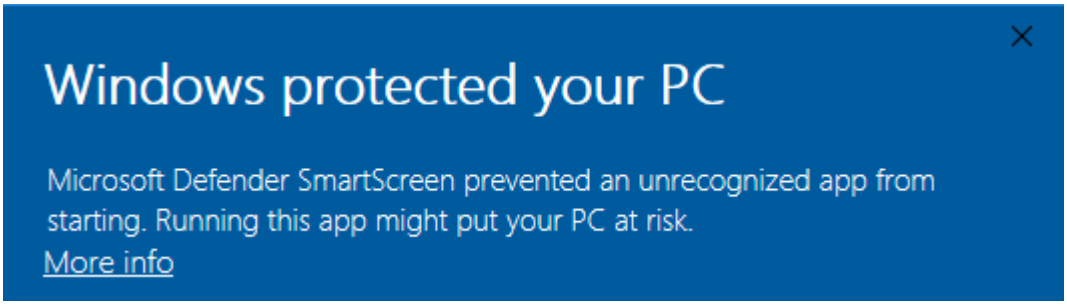
4.1.1 安装 EzBlock 镜像

在组装机器人之前，按照下面的步骤给树莓派的 Micro SD 卡烧录 EzBlock 系统。

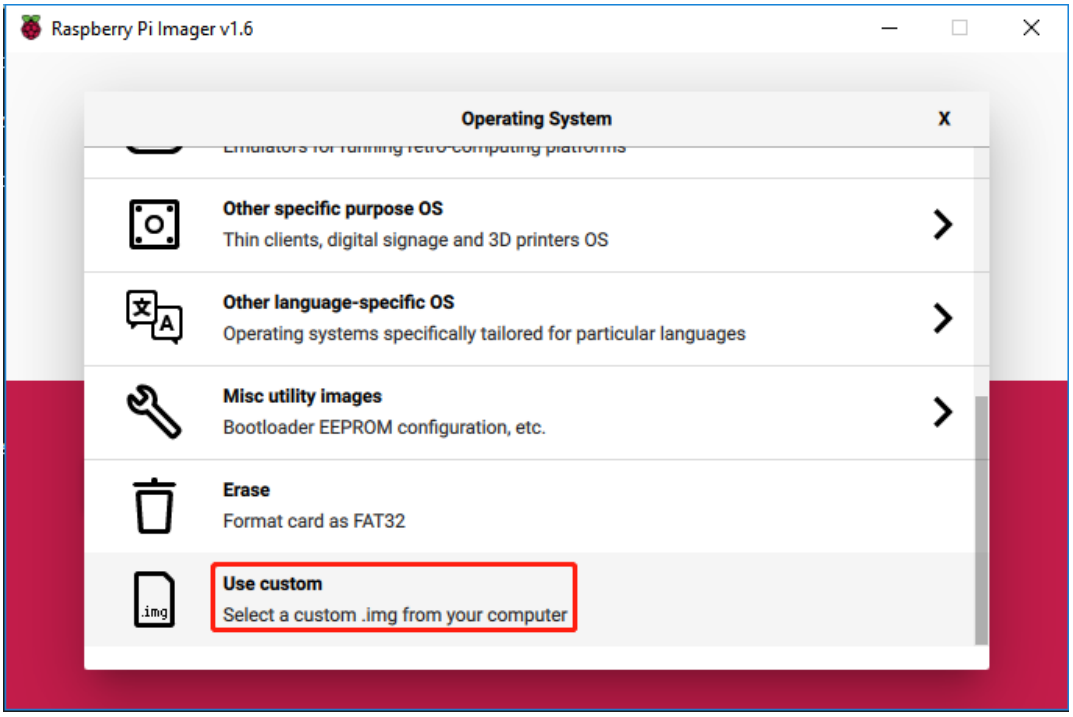
1. 树莓派开发了一个图形 SD 卡写入工具，适用于 Mac OS、Ubuntu 18.04 和 Windows，对于大多数用户来说是最简单的选择，因为它会下载映像并将其自动安装到 SD 卡。访问下载页面：https://www.raspberrypi.org/software/。单击与您的操作系统匹配的 Raspberry Pi Imager 链接，下载完成后，单击它以启动安装程序。



2. 当您启动安装程序时，您的操作系统可能会尝试阻止您运行它。例如，在 Windows 上，我收到以下消息：如果出现此消息，请点击 **更多信息**，然后点击 **仍然运行**，然后按照说明安装 Raspberry Pi Imager。

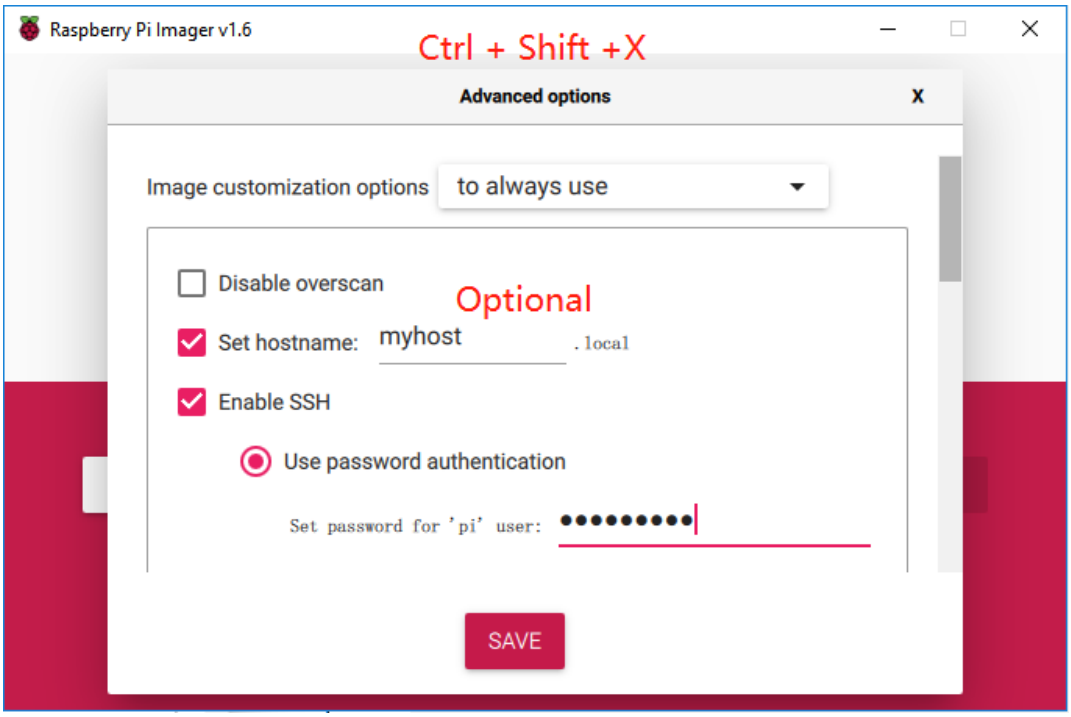


3. 将 SD 卡用读卡器插入计算机或笔记本电脑的 SD 卡插槽。
4. 下载 EzBlock 镜像
   - 天翼网盘：链接：https://cloud.189.cn/t/QNrEJnEjYNRz
   - 百度网盘：链接：https://pan.baidu.com/s/1ku1VoukCebChq9-OzkHf_g?pwd=ezbl，提取码：ezbl。
   - 由于文件超过 1G，需要在电脑上下载客户端之后才能下载文件。
5. 在 Raspberry Pi Imager 中选择刚下载的 EzBlock 镜像。



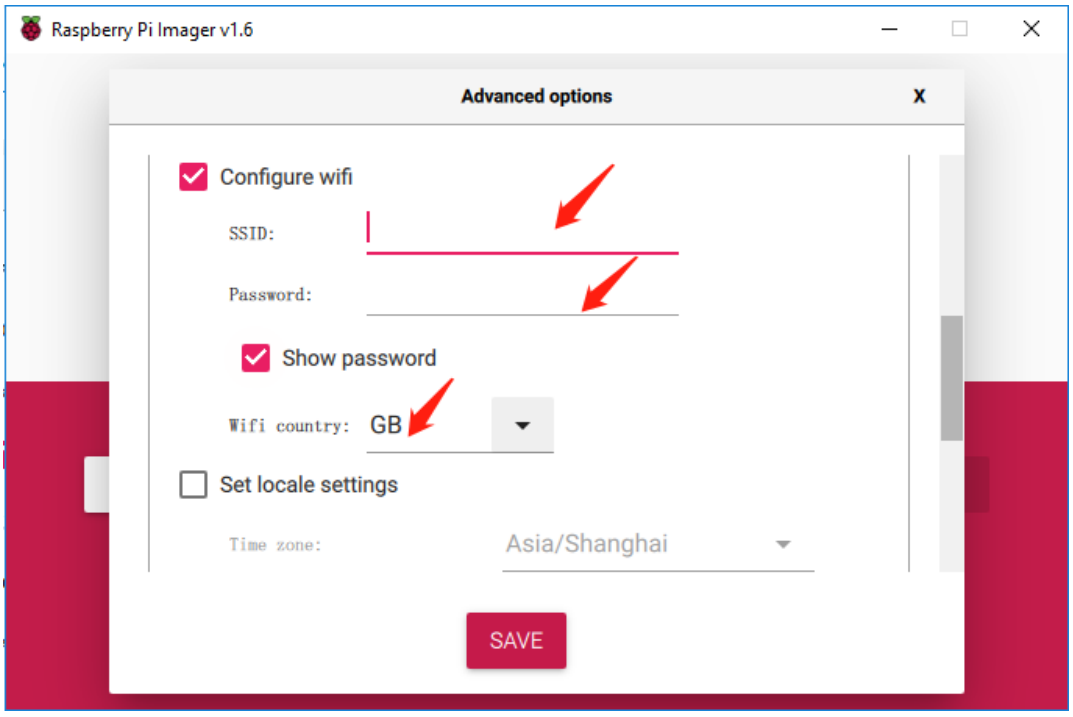
6. 按下 **Ctrl+Shift+X** 或者点击 **设置**按钮来打开 **Advanced options** 页面来设置 `hostname` 和启动 `SSH`。你可以选择“always use this image customization options”（始终使用该定制选项）。

备注： `hostname` 是让你在使用网页版 *EzBlock*, 可以用它来连接到你的产品，你也可以不设置。

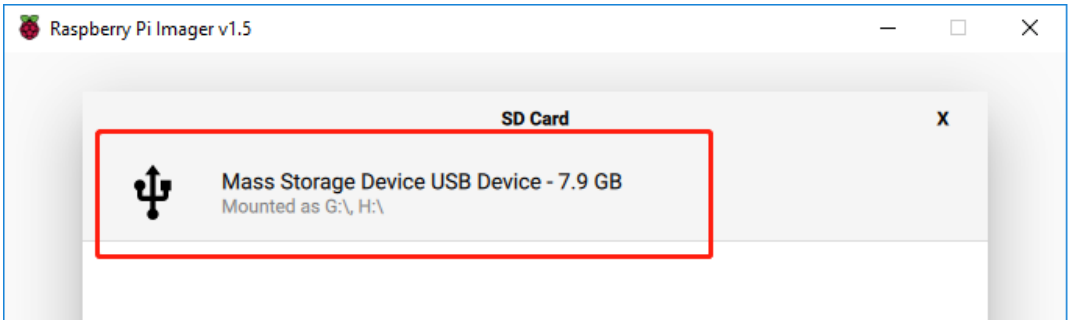


7. 下拉到底完成 WiFi 配置，然后点击 **SAVE**。

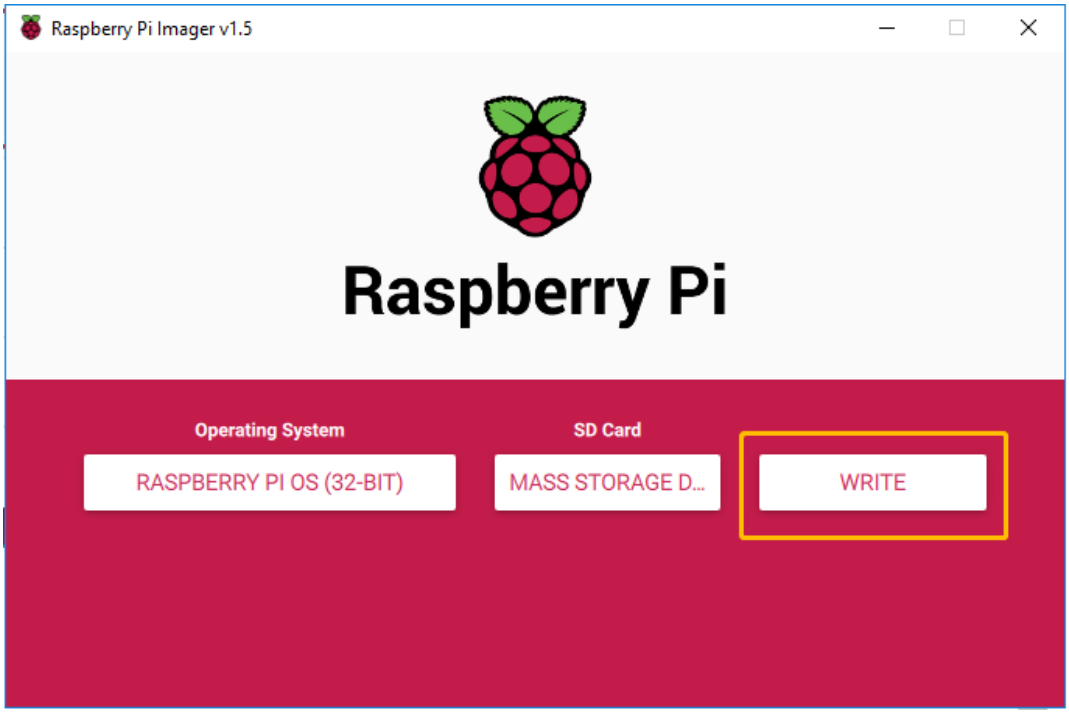
备注：wifi country 选择 CN



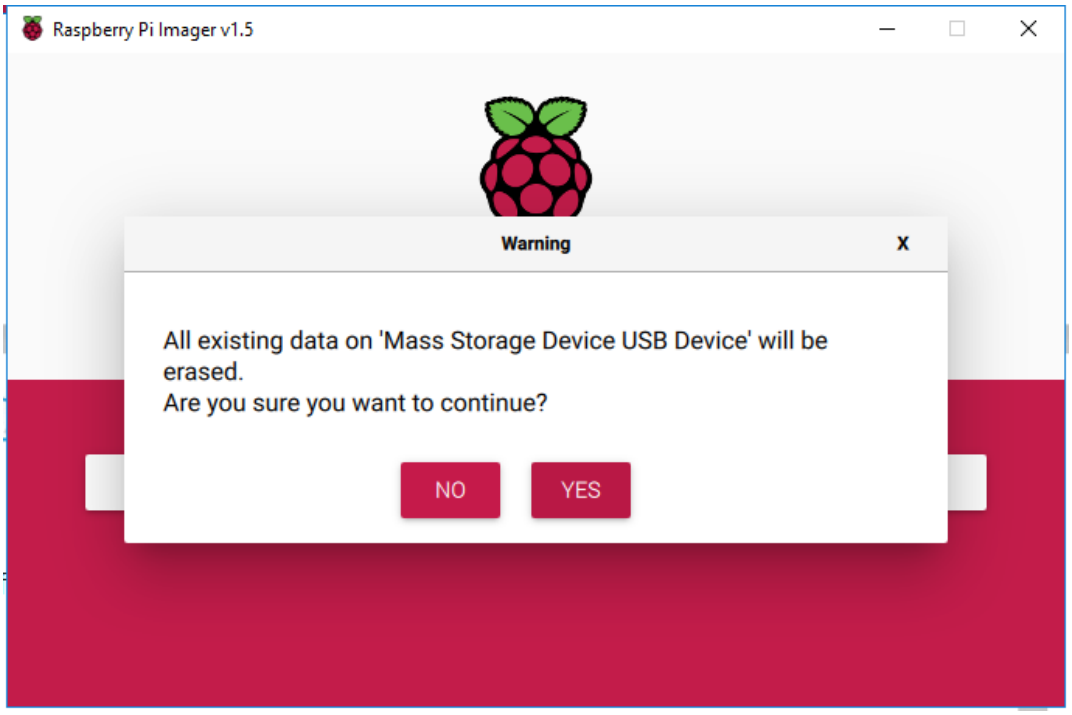
8. 选择您正在使用的 SD 卡。



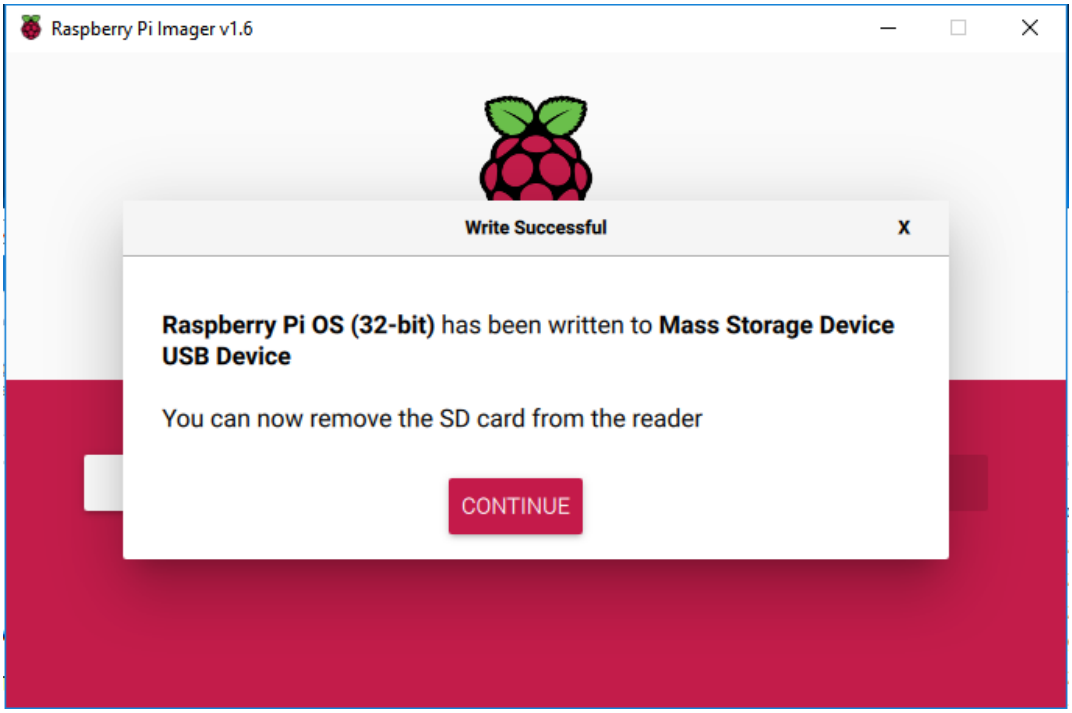
9. 单击 **WRITE** 按钮。



10. 如果您的 SD 卡上当前有任何文件，您可能希望先备份这些文件以防止永久丢失它们。如果没有要备份的文件，请单击 **YES**。



11. 等待一段时间后，会出现如下窗口，代表写入完成。现在你就可以将读卡器从电脑中拔出，并取出 SD 卡插入到树莓派上。

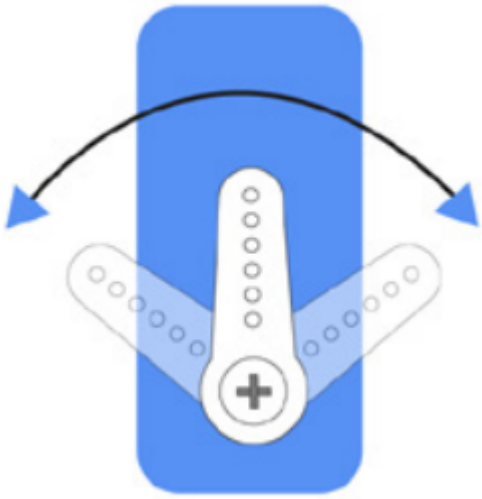


4.1.2 舵机调零

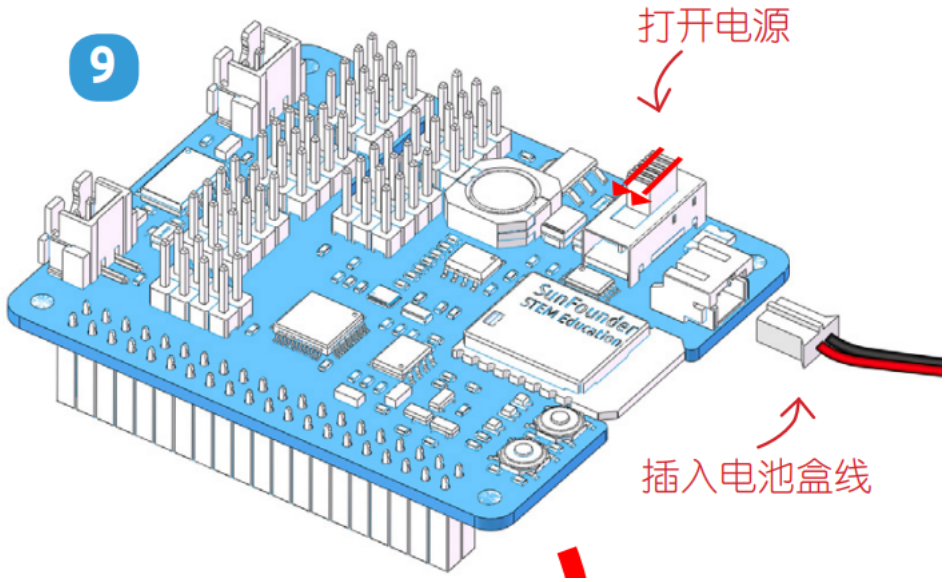
伺服器的角度范围是-90~90，但出厂时设置的角度是随机的，可能是 0°，也可能是 45°；如果我们直接用这样的角度组装，运行代码后机器人将处于混乱状态，更糟糕的是，可能导致伺服器阻塞并烧毁。

因此，我们需要将所有伺服机的角度设置为 0°，然后再进行安装，这样伺服机的角度就处于中间位置，无论往哪个方向转。

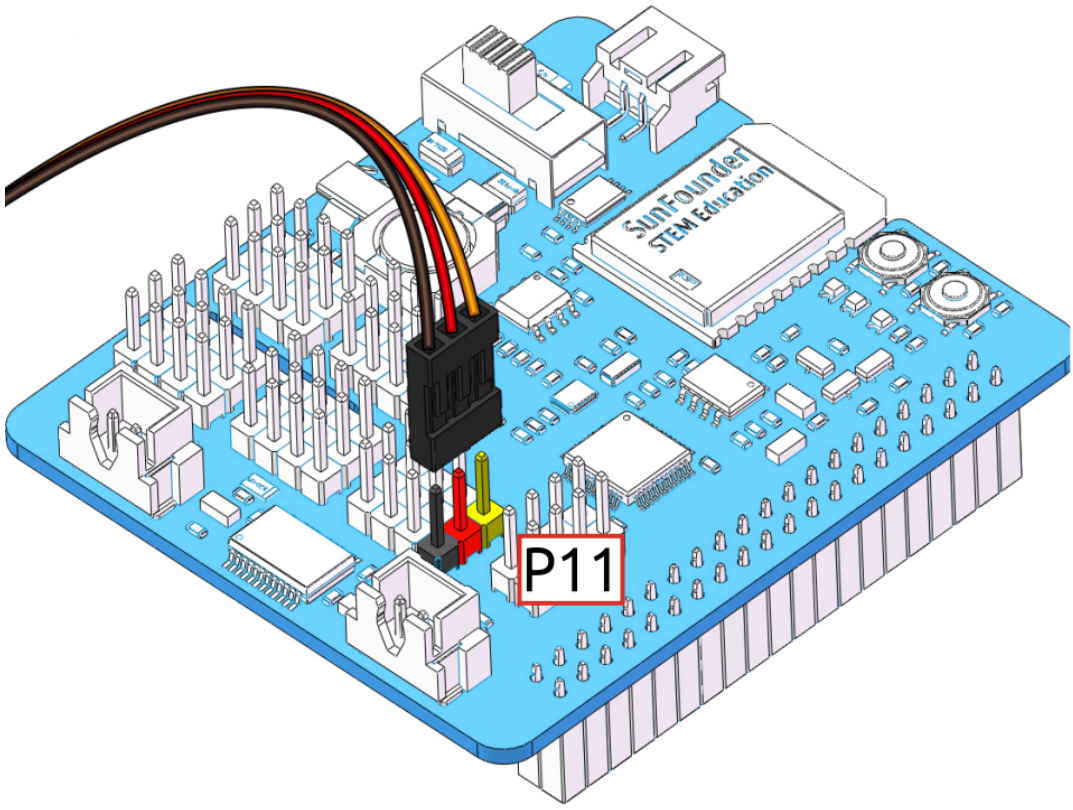
1. 为确保伺服器已正确设置为 0°，首先将伺服臂插入伺服轴，然后轻轻旋转摇臂到另一个角度。这个伺服臂只是让您清楚地看到伺服器正在旋转。



2. 按照组装折页上的指示，插入电池盒电缆并将电源开关拨到 ON。等待 1-2 分钟，Raspberry Pi 启动成功时会有声音提示。

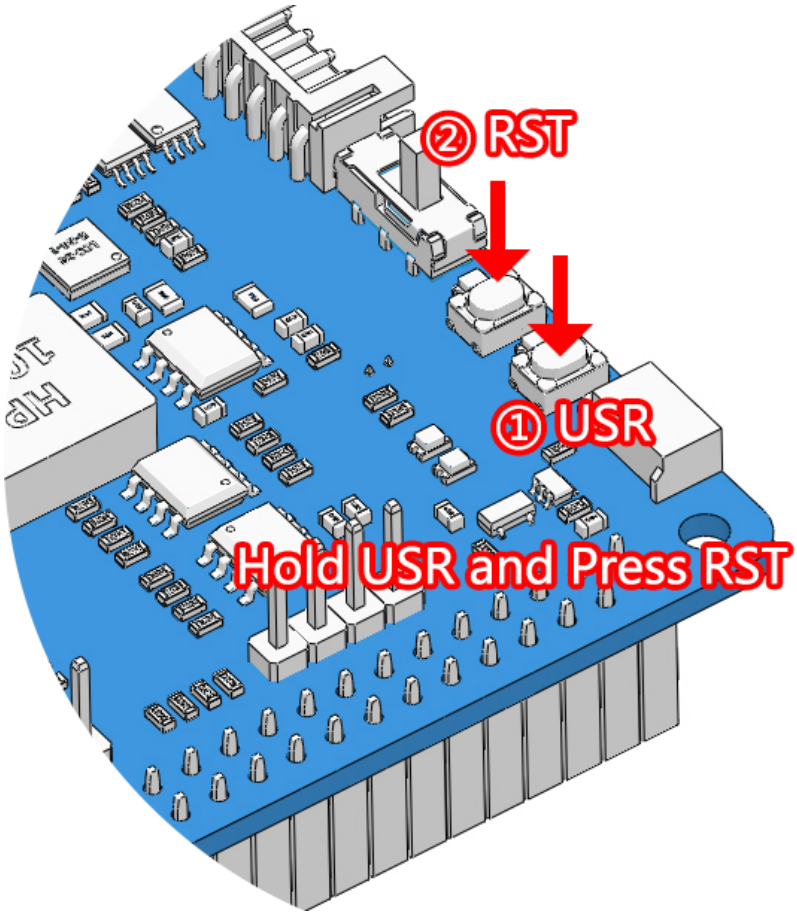


3. 接下来，按照以下方式将伺服电缆插入 P11 端口。



4. 长按 **USR** 键，然后按 **RST** 键执行系统内的伺服归零脚本。当您看到伺服臂旋转到某个位置时（这是 0° 位置，是一个随机位置，可能不是垂直或平行的。），表明程序已运行。

> **备注：** 这一步只需要执行一次；之后，只需插入其他伺服线，它们会自动归零。



5. 现在，拔下伺服臂，确保伺服线保持连接，且不要关闭电源。然后按照纸质组装说明继续组装。

备注:

- 在用伺服螺丝固定此伺服前，不要拔下此伺服电缆，固定后可以拔下。
- 通电状态下不要旋转伺服，以避免损坏；如果伺服轴插入的角度不对，拔出伺服并重新插入。
- 在组装每个伺服前，需要将伺服电缆插入 P11 并打开电源，将其角度设置为 0°。
- 如果您稍后使用 EzBlock APP 向机器人下载程序，此归零功能将被禁用。

4.2 下载 EzBlock Studio

扫描下方二维码，下载 EzBlock Studio APP。

备注:

- 对于安卓手机，会提示用浏览器打开。下载成功后，根据提示允许浏览器安装该应用
- 对于 iPhone 手机，会打开 APP Store 来安装该应用。
- 对于笔记本和电脑可以使用网页版 *EzBlock*。

4.3 快速配置

在你第一次用 EzBlock 对你的机器人进行编程时，会有一个快速配置的过程。

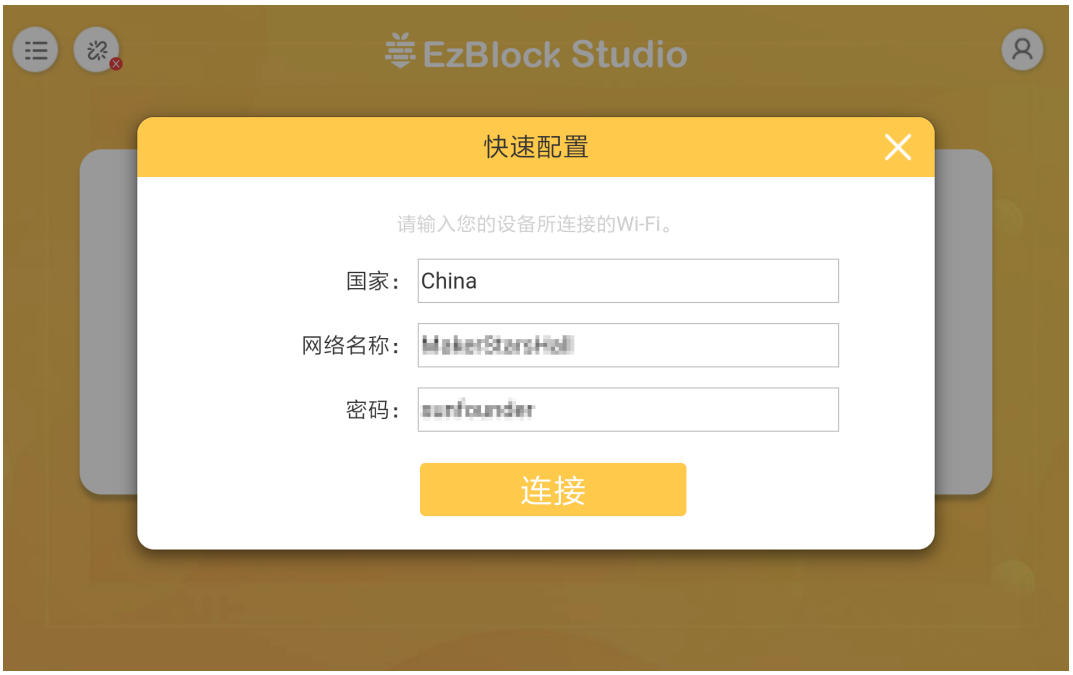
1. 打开 EzBlock Studio，会弹出一个空设备列表的窗口。您需要在打开产品电源的同时打开移动设备的蓝牙，才会出现产品编号。

```
ezb-RaspberryPi(2A:05:12:00:00:CB)
```

2. 点击右上角的 **完成**，稍等片刻，会提示 **连接成功**。此时您需要点击 **确定**来快速配置您的产品。

3. 输入您的 Wi-Fi 帐户和密码。

> **备注：** 如果您已经提前配置了 Wi-Fi，那么 **快速配置**这一步就不会出现，直接进入下一步 **设置名称**。

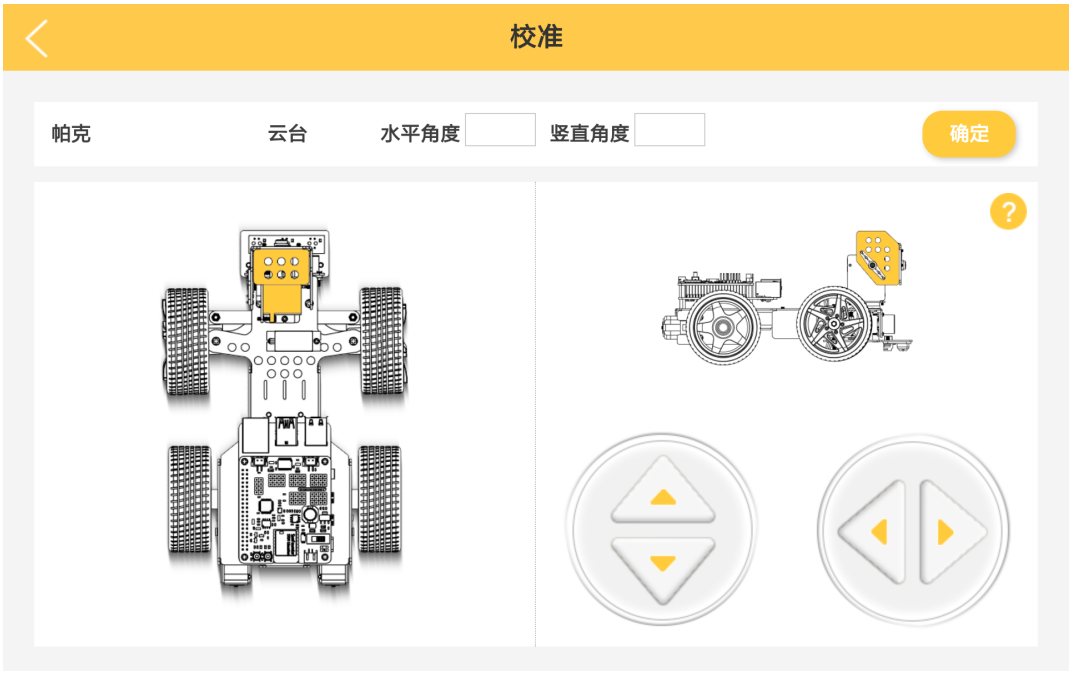


4. 选择与您匹配的产品。

5. 为您的产品输入名称。

6. 如果您的产品需要校准，会有提示告诉您可以通过点击 **立即校准**进入校准页面。如果不需要，则弹出窗口消失并返回主页。

7. 每个产品的校准页面都不一样，但是都有提示需要校准的部分。您可以点击相应的部分，然后参考 **校准帮助**进行校准。校准完成后，点击 **确认**。



4.4 打开或新建项目

配置完成之后，你就可以开始给你的机器人编写程序，我们已经将所有示例都已上传到 **示例**页面，你可以直接运行或打开相应的项目。或者根据课程中的提示，新建项目后，通过简单的拖拽来完成相应项目的编程。

打开项目

1. 在首页，点击 **示例**，进入示例页面。如果您只需要简单地测试这些示例，您只需点击 **运行**即可让您的产品工作。如果要查看和修改里面的代码，那么就需要点击 **编辑**进入编程页面。

2. 进入编程页面之后，将代码修改之后，点击 **下载/运行**按钮，就能看到你的机器人做出相应的效果。可点击 **运行/暂停**按钮来停止运行。

新建项目

1. 点击 **新建项目**按钮。

2. 填写项目名称。

3. 根据每个项目的提示，编写程序。

4. 点击 **下载/运行**按钮，就能看到你的机器人做出相应的效果。可点击 **运行/暂停**按钮来停止运行。

EzBlock 项目

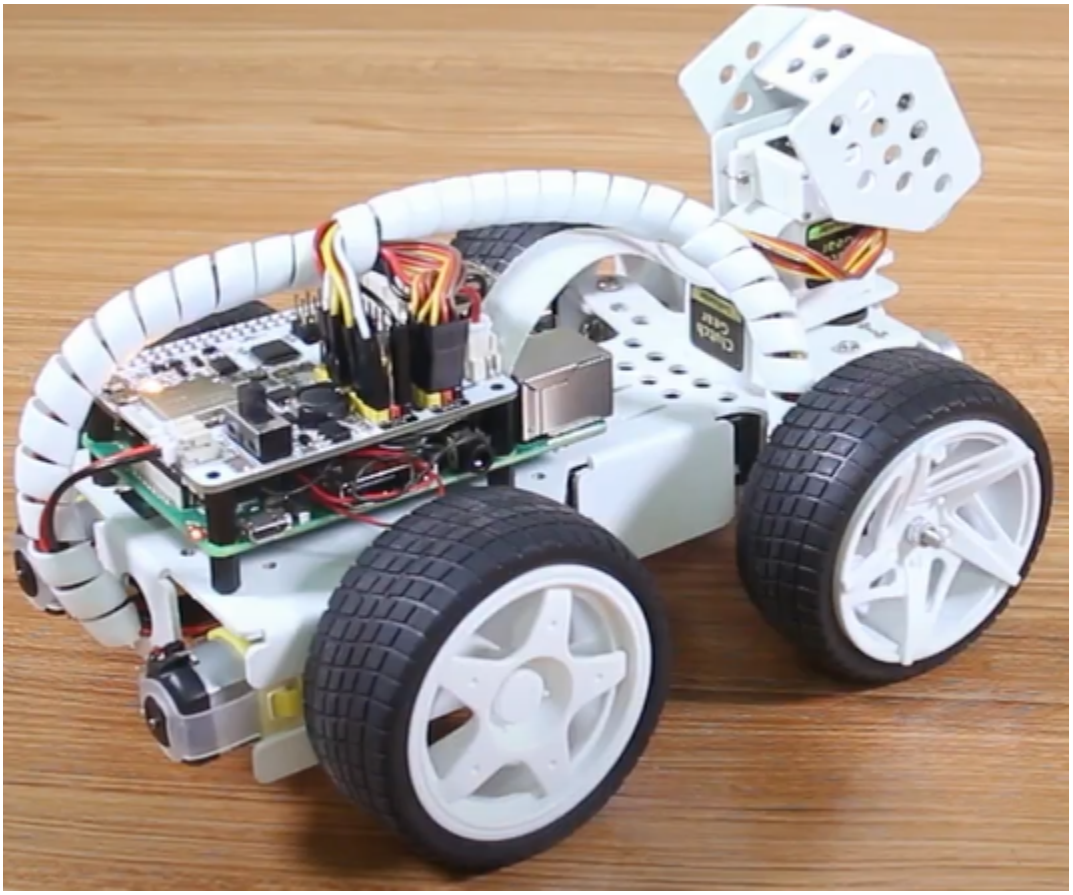
本节从帕克的基本编程功能开始，一直到在 EzBlock Studio 中创建更高级的程序。每个教程都包含介绍新功能的提示，使用户能够编写相应的程序。在示例部分也有完整的参考代码，可以直接使用。我们建议在不使用示例部分的代码的情况下尝试编程，并享受克服挑战的快乐体验

基础

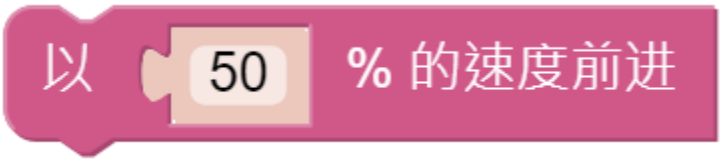
4.5 移动

备注： 在首页，点击 **示例**，进入示例页面，可以直接打开预设好的示例。如果您只需要简单地测试这些示例，您只需点击 **运行**即可让您的产品工作。

第一个项目教授如何为帕克编程运动动作。在这个项目中，程序会告诉帕克依次执行五个动作：“前进”、“后退”、“左转”、“右转” 和 “停止”。



提示



该模块将使帕克以基于可用功率百分比的速度前进。在下面的示例中，“50”是功率的 50%，或者说是半速。

该模块将使帕克以基于可用功率百分比的速度向后移动。

该块调整前轮的方向。范围是“-45”到“45”。在下面的示例中，“-30”表示车轮将向左转 30°。

此块将导致命令之间的定时中断，基于毫秒。在下面的示例中，帕克在执行下一个命令之前将等待 1 秒（1000 毫秒）。

该块将使帕克完全停止。

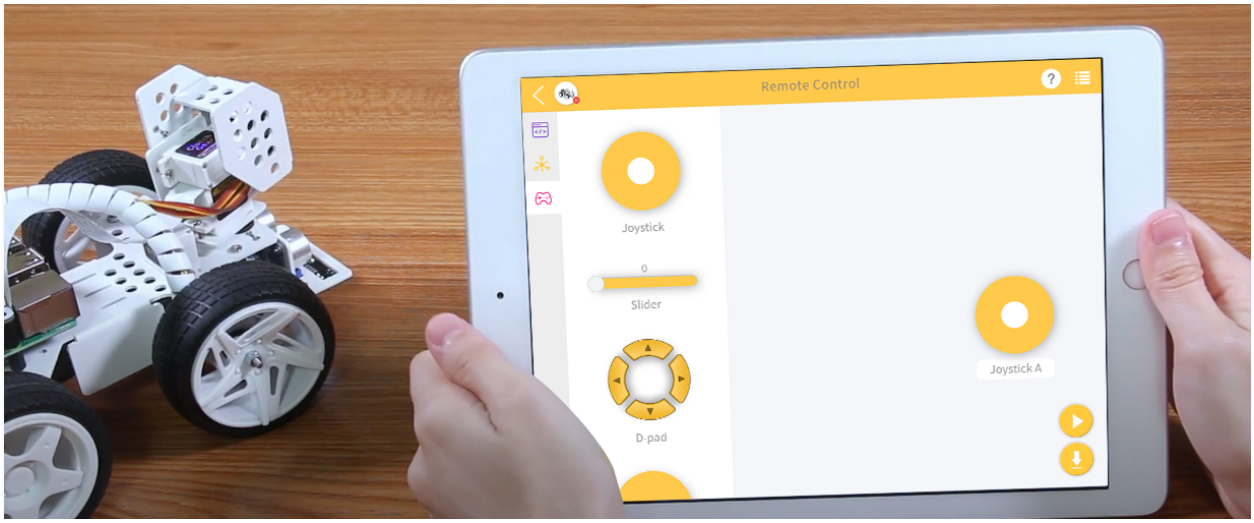
示例

备注： 代码完成后，点击右下角的“下载 & 运行”按键，让帕克动起来。

开始
循环
以 50 % 的速度前进
延时 1000
以 50 % 的速度后退
延时 1000
以 50 % 的速度前进
调整转向角度为 -30
延时 1000
以 50 % 的速度前进
调整转向角度为 30
延时 1000
调整转向角度为 0
停下
延时 2000

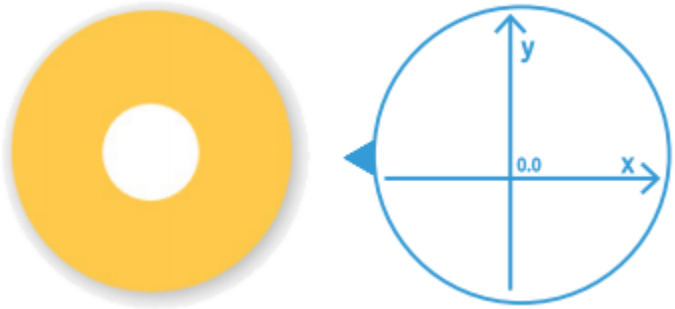
4.6 远程遥控

该项目将教授如何使用操纵杆小部件远程控制帕克。注意：从远程控制页面拖放操纵杆小部件后，使用“映射”功能校准操纵杆 X 轴和 Y 轴读数。



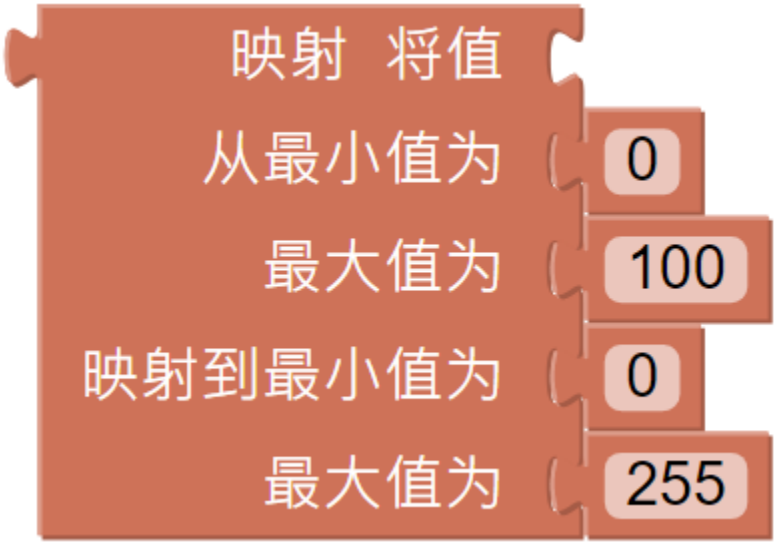
提示

要使用远程控制功能，请从主页面左侧打开远程控制页面。



将操纵杆拖动到远程控制页面的中央区域。切换中心的白点，并在任何方向轻轻拖动将产生 (X,Y) 坐标。X 轴或 Y 轴的范围默认为 “-100” 到 “100”。切换白点并将其直接拖动到操纵杆的最左侧将导致 X 值为 “-100”，Y 值为 “0”。

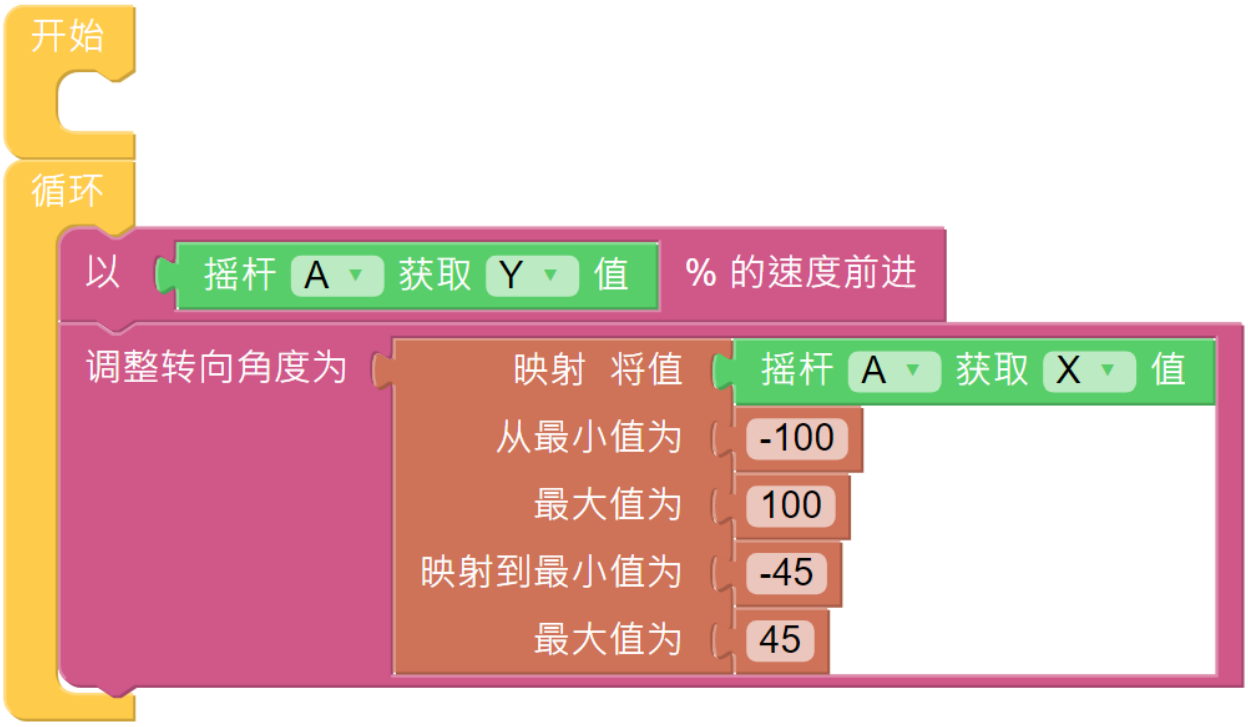
在遥控器页面拖放一个小部件后，积木块栏会出现一个新的分类-远程遥控，上面会更新出新的块。此块读取远程控制页面中的操纵杆值。您可以单击下拉菜单切换到 Y 轴读数。



映射值块可以将数字从一个范围重新映射到另一个范围。如果范围设置为 0 到 100，并且映射值为 50，则它位于范围的 50% 位置，即“50”。如果范围设置为 0 到 255，并且映射值为 50，则它位于范围的 50% 位置，即“127.5”。

示例

备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.



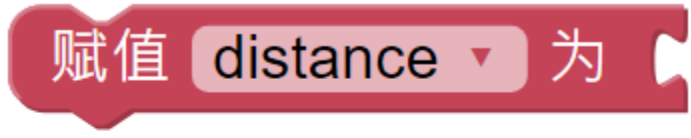
代码运行后，进入到远程控制页面，就可以使用摇杆来控制 PiCar-X 的移动。

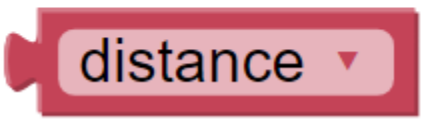
4.7 测试超声波模块

帕克内置超声波传感器模块，可用于避障和自动物体跟随实验。在本课中，模块将读取以厘米为单位的距离（24 厘米 = 1 英寸），并在调试窗口中打印结果。

提示

超声波获取距离块将读取从帕克与正前方障碍物的距离。

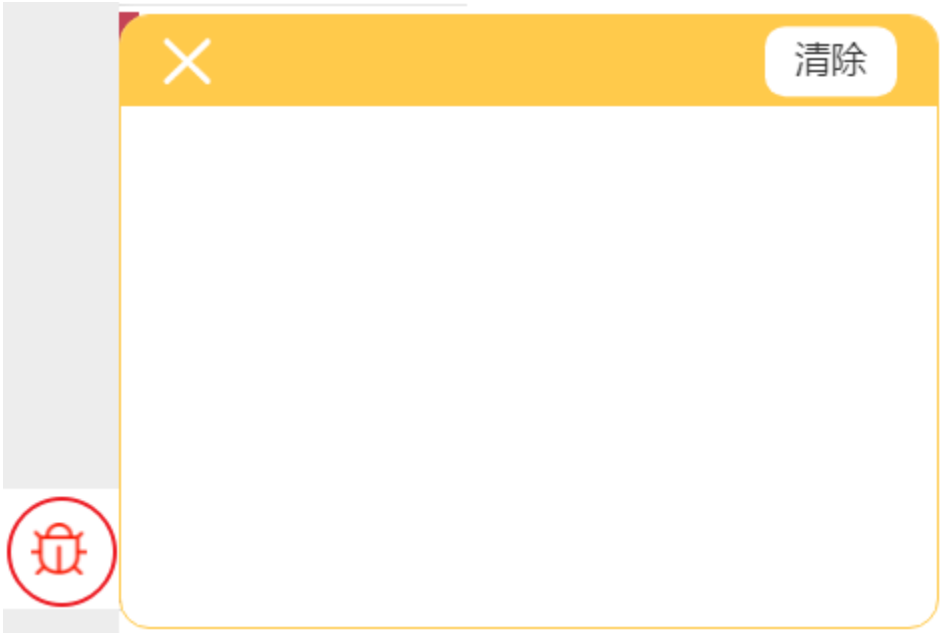




该程序使用 **变量**进行了简化。例如，当程序中有多个函数，每个函数都需要使用到障碍物的距离时，可以使用 **变量**向每个函数报告相同的距离值，而不是每个函数单独读取相同的值.

使用变量需要先在 **变量**类别里点击 **创建变量...** 按钮。

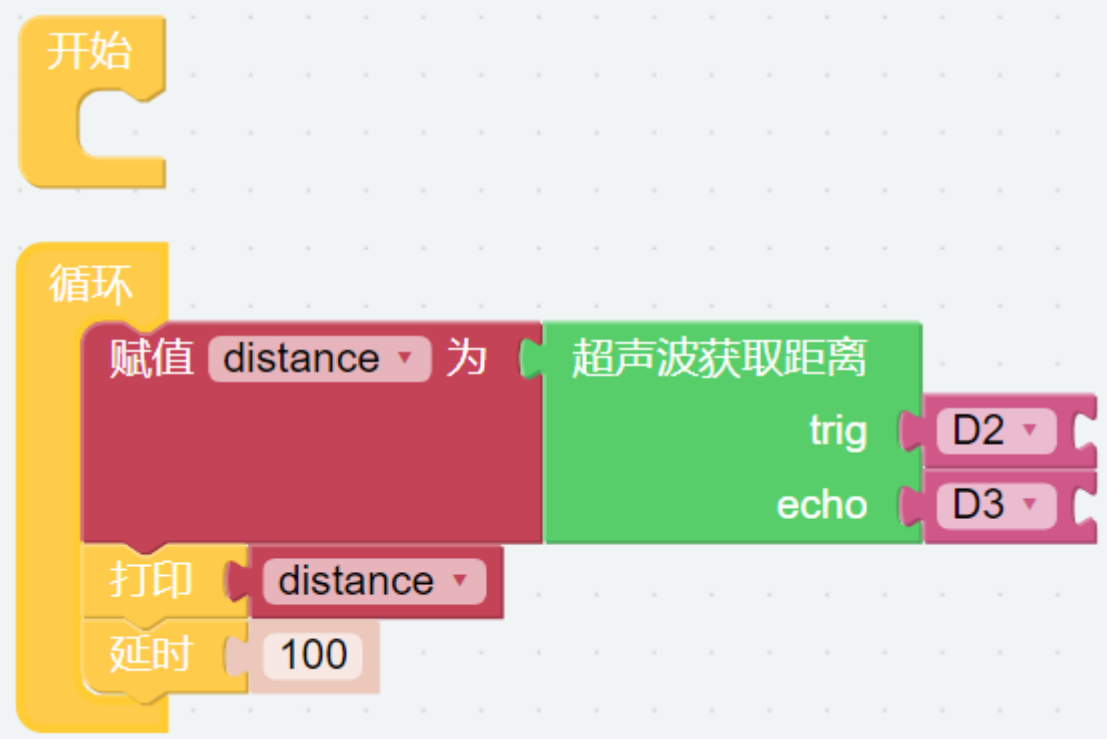
打印功能可以打印变量、文本等数据，方便调试。



代码运行后，通过单击左下角的 **调试**图标启用调试监视器。

示例

备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.



代码运行后，你将看到检测的距离值显示在调试监视器中。

4.8 测试灰度模块

帕克包含一个灰度模块，用于实现线路跟踪、悬崖检测和其他有趣的实验。灰度模块具有三个检测传感器，每个传感器将根据传感器检测到的颜色深浅报告一个值。例如，读取纯黑色阴影的传感器将返回值“0”。

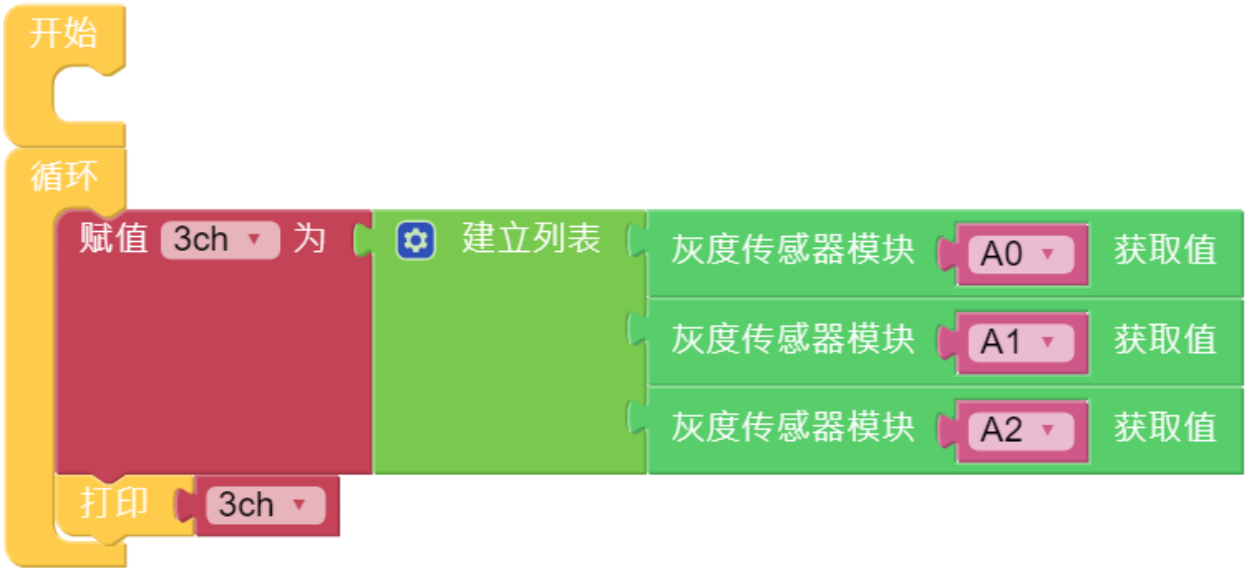
提示

使用 **灰度模块**块读取其中一个传感器的值。在上面的示例中，“A0”传感器是帕克最左侧的传感器。使用下拉箭头将传感器更改为“A1”（中心传感器）或“A2”（最右侧的传感器）。

该程序通过 **建立列表**块进行了简化。**列表**的使用方式与单个 **变量**的使用方式相同，但在这种情况下，**列表**比单个 **变量**更有效，因为 **灰度模块**将报告多个传感器值。**建立列表**块将为每个传感器创建单独的 **变量**，并将它们放入一个列表中。

示例

备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.



代码运行后，你将看到调试监视器中打印的 3 组值，越黑的环境，打印的值越小。

4.9 颜色检测

帕克是一款内置摄像头的自动驾驶汽车，它允许 EzBlock 程序利用物体检测和颜色识别代码。在本节中，EzBlock 将用于创建颜色检测程序。

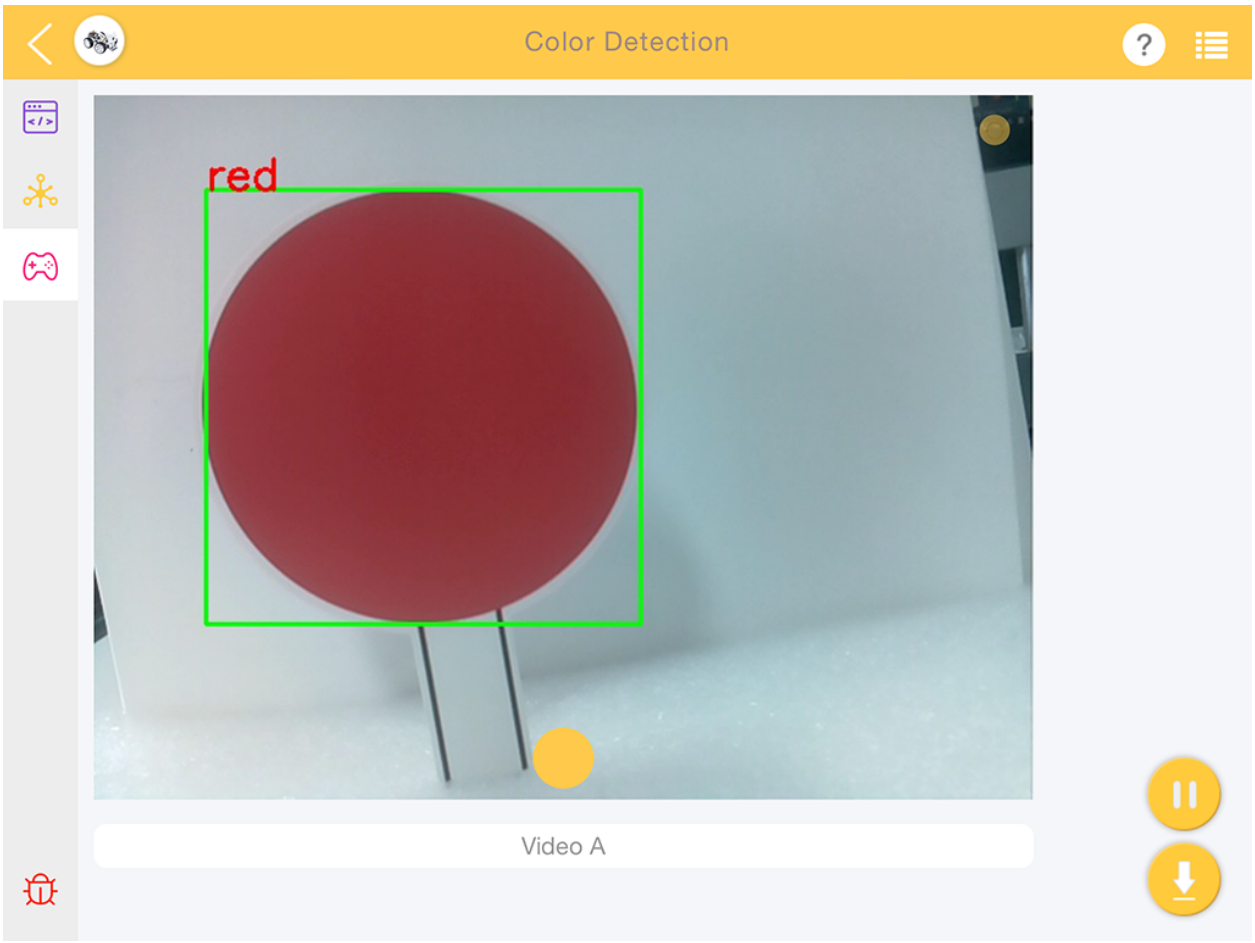
备注： 在尝试本部分之前，请确保树莓派相机的 FFC 电缆已正确且牢固地连接。

在这个程序中，EzBlock 首先会被告知待检测颜色的 HSV（Hue-Saturation-Value）空间范围，然后利用 OpenCV 对 HSV 范围内的颜色进行处理去除背景噪声，最后对匹配颜色进行框选。

EzBlock 包括帕克的 6 种颜色模型，“红色”、“橙色”、“黄色”、“绿色”、“蓝色” 和 “紫色”。色卡已在以下 PDF 中准备好，需可以下载下来打印。

- [PDF] 颜色卡

备注： 由于打印机碳粉或打印介质（如棕褐色纸）的不同，打印颜色可能与颜色模型的色调略有不同。这可能会导致不太准确的颜色识别。



提示

从远程控制页面拖动视频小部件，它将生成一个视频监视器。

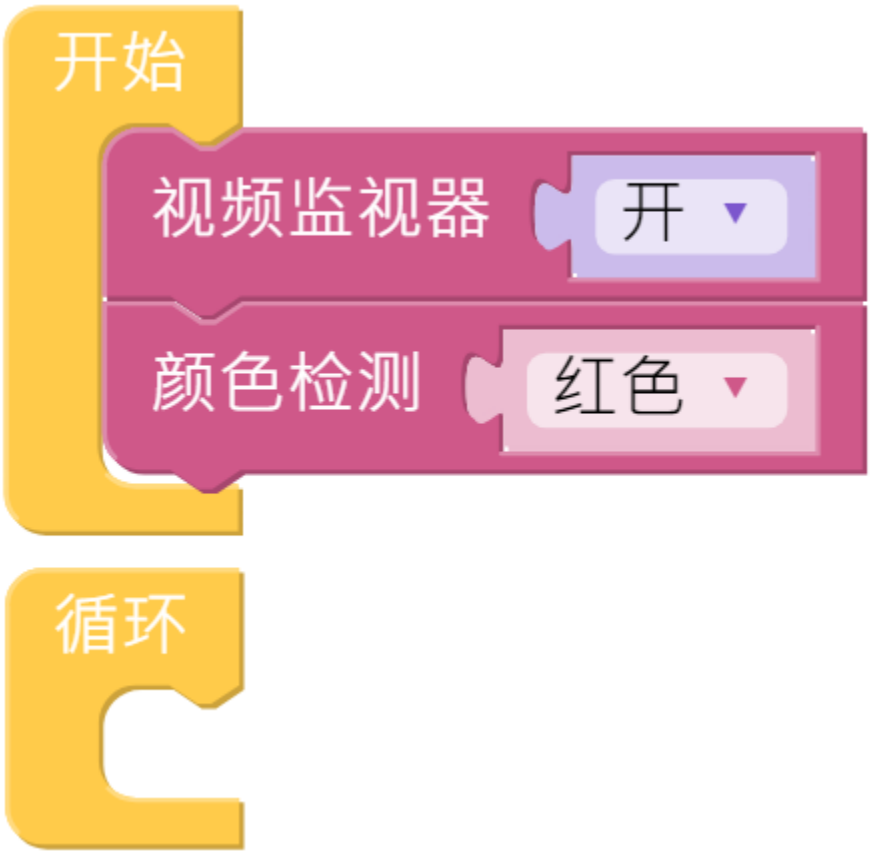
点击视频监视器将会出现属性框，你可以在这里调节它的大小。

通过将 **视频监视器**块设置为 **开**来启用视频监视器。注意：将 **视频监视器**设置为 **关**将关闭监视器，但对象检测仍然可用。

使用 **颜色检测**块来启用颜色检测。注意：一次只能检测一种颜色。

示例

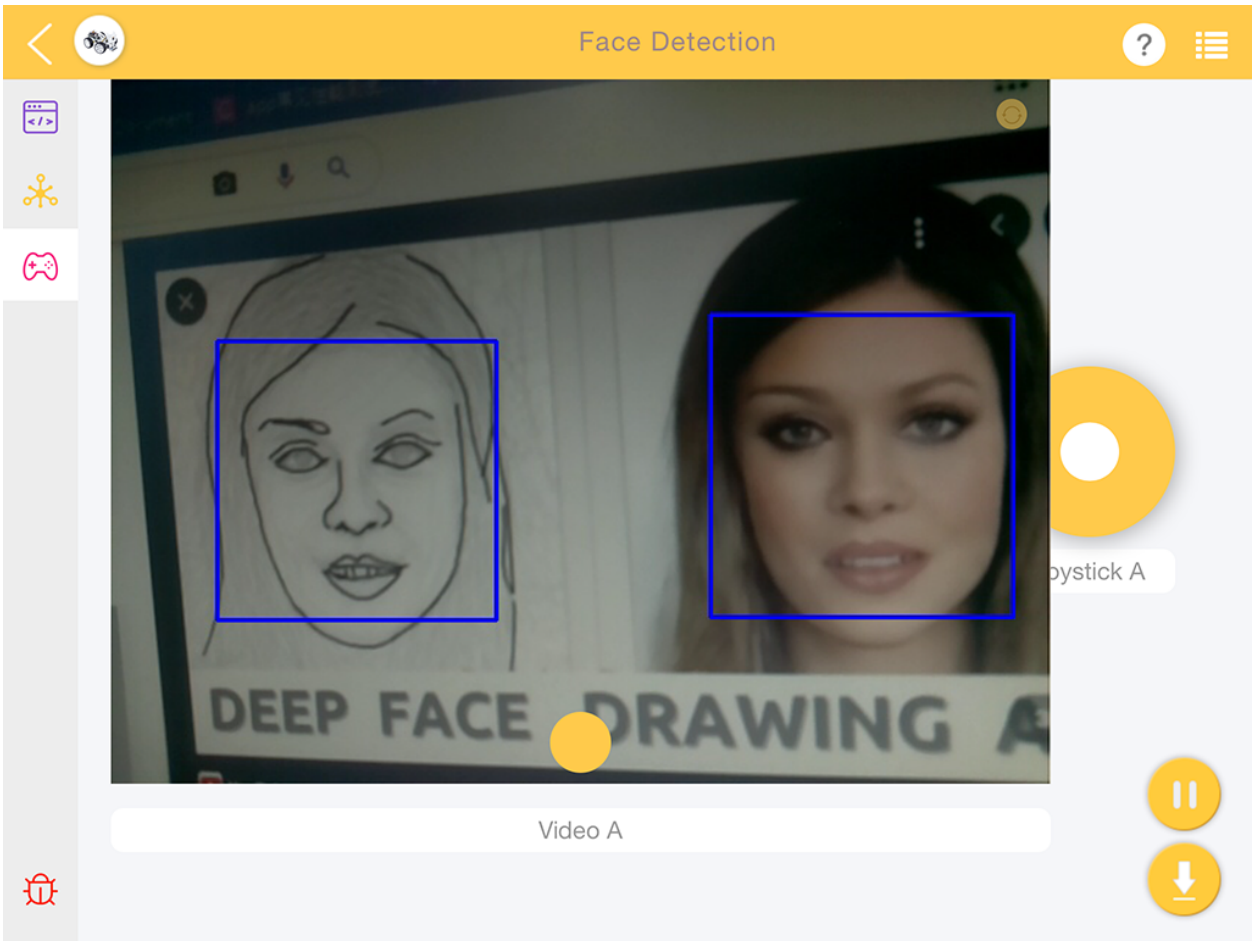
备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.



代码运行后，进入到远程控制界面，你将看到摄像头拍摄的画面，并且将红色物体框选出来。

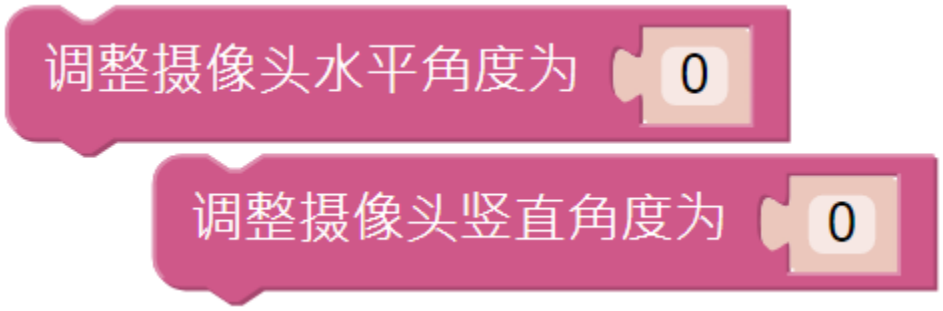
4.10 人脸检测

除了颜色检测外，帕克还包含人脸检测功能。在以下示例中，操纵杆小部件用于调整相机的方向，并且调试监视器中将显示面部数量。



提示

将 **人脸检测**设置为 **开**以启用面部检测。



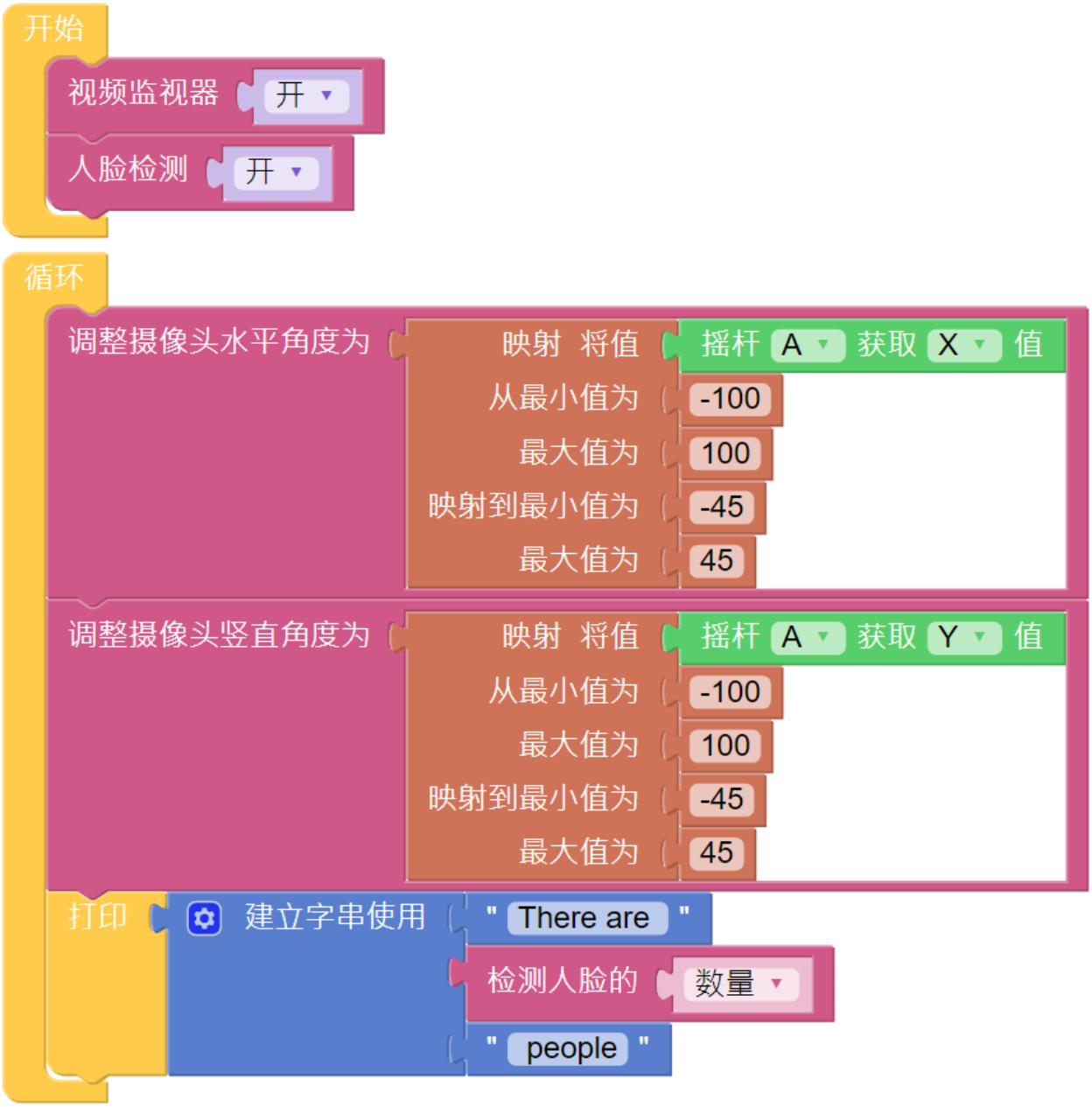
这两个块用于调整云台相机的方向，类似于远程遥控 教程中的驾驶帕克。随着值的增加，相机将向右或向上旋转，减小的值将向右或向下旋转相机。

图像检测结果通过 **检测人脸**块给出。使用下拉菜单选项在从图像检测功能读取坐标、大小或结果数量之间进行选择。

使用 **建立字串使用**块来打印 **文本**和 **检测人脸**数据的组合。

示例

备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.

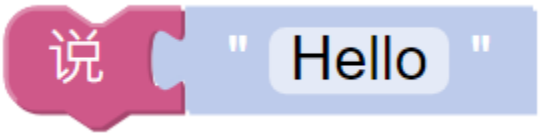


代码运行后，你将看到摄像头拍摄的画面；若 PiCar-X 拍摄到人，它将框选出来。

4.11 音效

帕克内置扬声器，可用于音频实验。EzBlock 允许用户输入文字让帕克说话，或制作特定的声音效果。在本教程中，帕克将使用 **重复多次**函数，在 3 秒倒计时后发出枪声。

提示



使用 **说**块和 **文本**块来写一个句子让帕克说。**说**块可用于文本或数字。

这是一个 **数字**块。

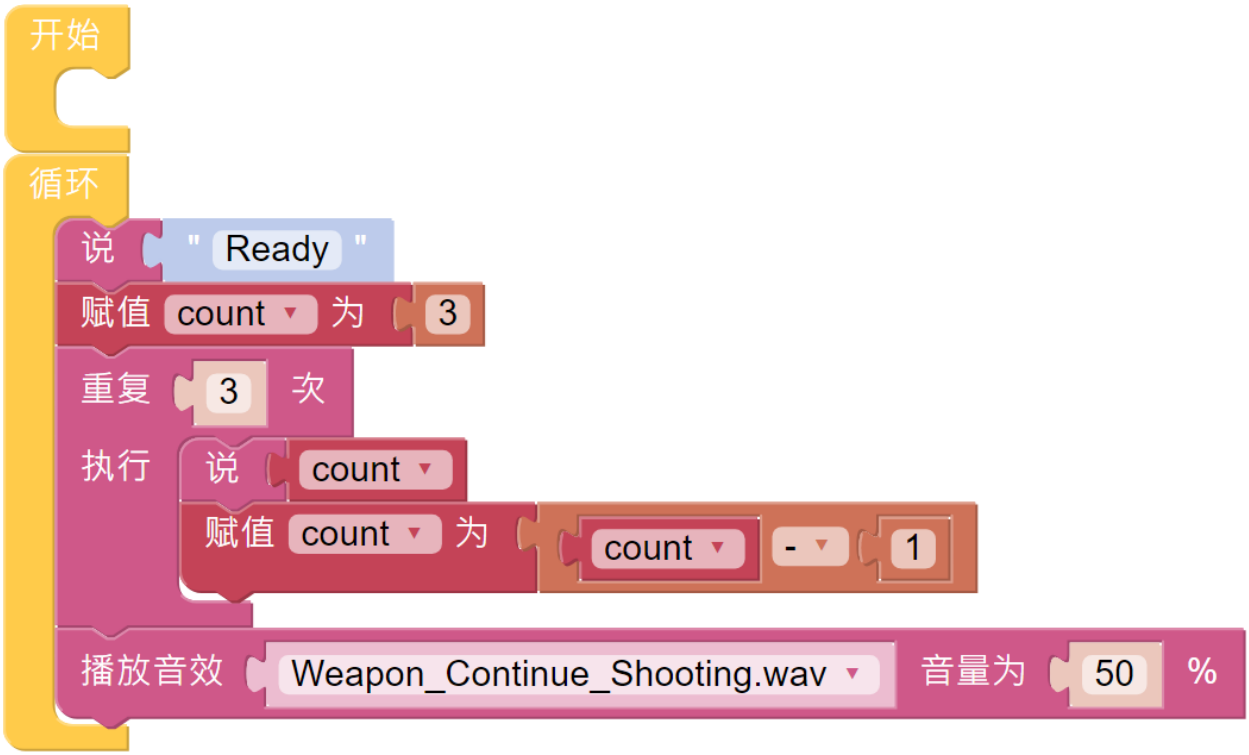
使用 **重复**块将重复执行相同的语句，从而减少代码的长度。

数学运算块可以执行典型的数学函数，例如“+”、“-”、“x”和“÷”。

播放音效块可以播放预设音效，例如警笛声、枪声等。音量范围可设置为 0 到 100。

示例

备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.



4.12 背景音乐

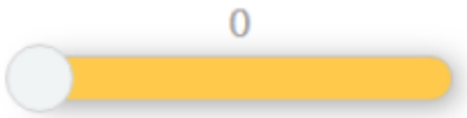
除了对帕克进行编程以播放音效或文字转语音 (TTS) 外，帕克还会播放背景音乐。该项目还将使用 **滑条**小部件来调整音乐音量。

关于 EzBlock 的遥控功能的详细教程，请参考远程遥控 教程。

提示

播放背景音乐块将需要添加到 **开始**功能。使用下拉菜单选择不同的背景音乐供帕克播放。

设置背景音乐音量块将在 0 到 100 的范围内调整音量。



从 **远程遥控**页面拖动 **滑条**栏以调整音乐音量。

滑条获取值块将读取滑块值。上面的示例选择了滑块“A”。如果有多个滑块，请使用下拉菜单选择合适的滑块。

示例

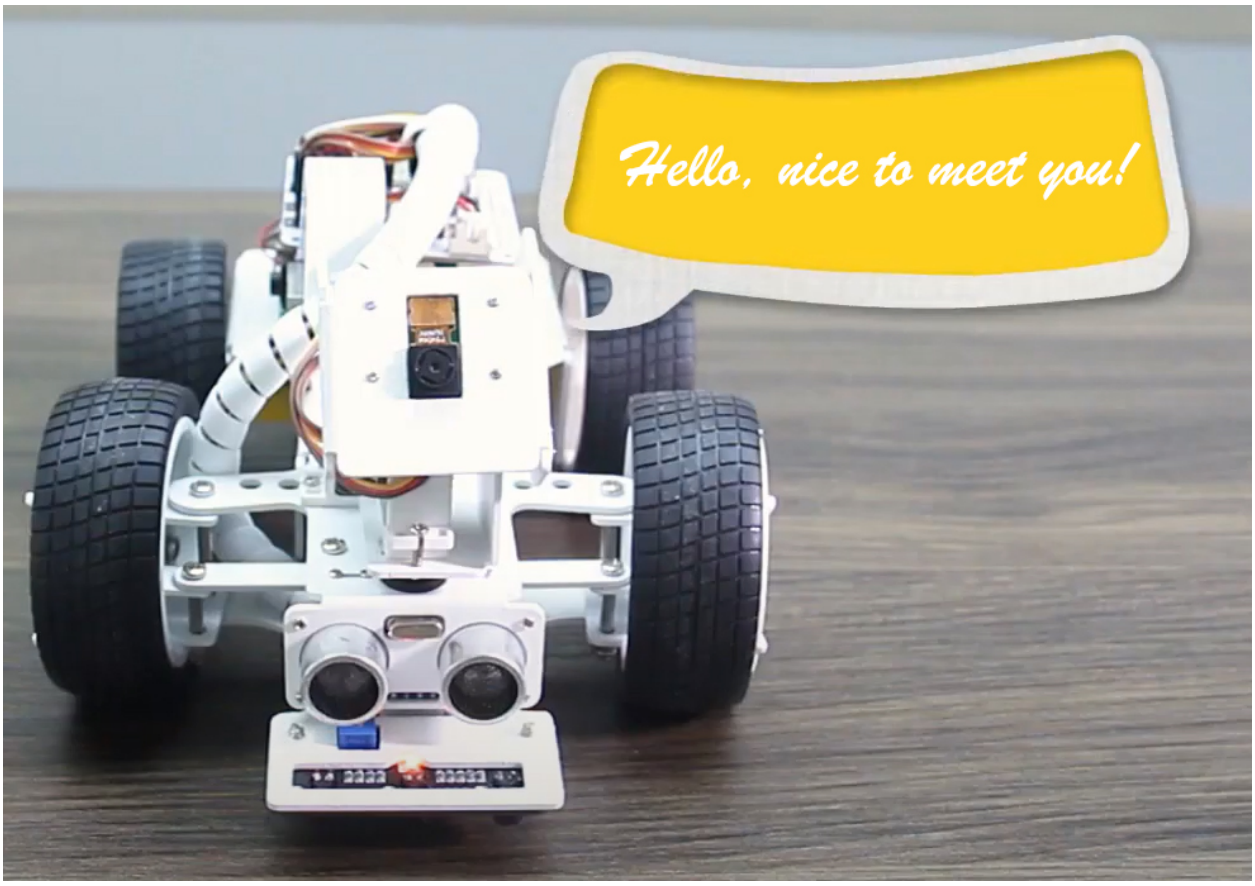
备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.

代码运行后，将播放一段背景音乐，你可以用远程控制页面的滑条来调节音量。

进阶

4.13 问好

这个项目将结合前面项目的几个功能。帕克运动将被远程控制，帕克的相机将通过使用两个操纵杆控制器进行远程控制。当帕克认出某人的脸时，它会礼貌地点点头，然后说“你好!”。



提示

如果-执行块使在 **如果**的条件判断为真时执行些功能。

条件语句块需要与 **如果-执行**块结合使用。条件可以是“=”、“>”、“<”、“≥”、“≤”或“≠”。

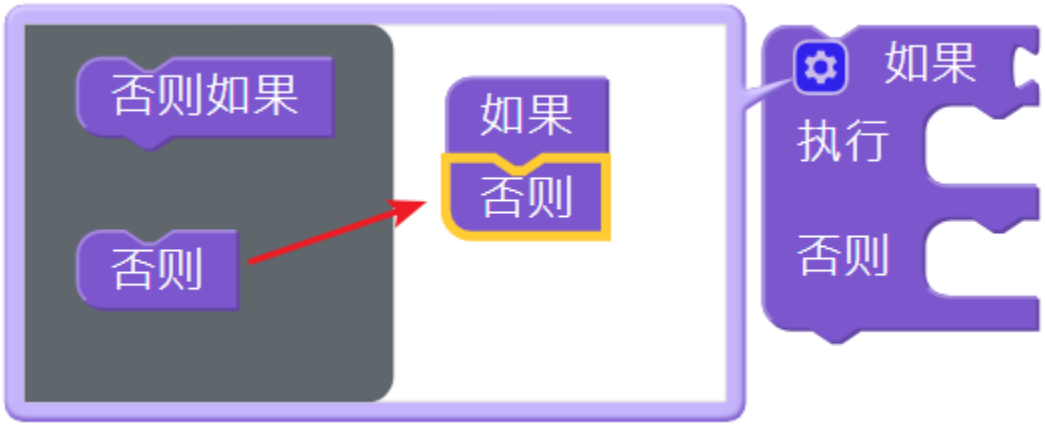
示例

备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.

4.14 音乐车

该项目将把帕克变成一辆音乐车，可以在您家中穿梭，播放欢快的音乐。该项目还将展示帕克如何通过内置超声波传感器避免撞击墙壁。

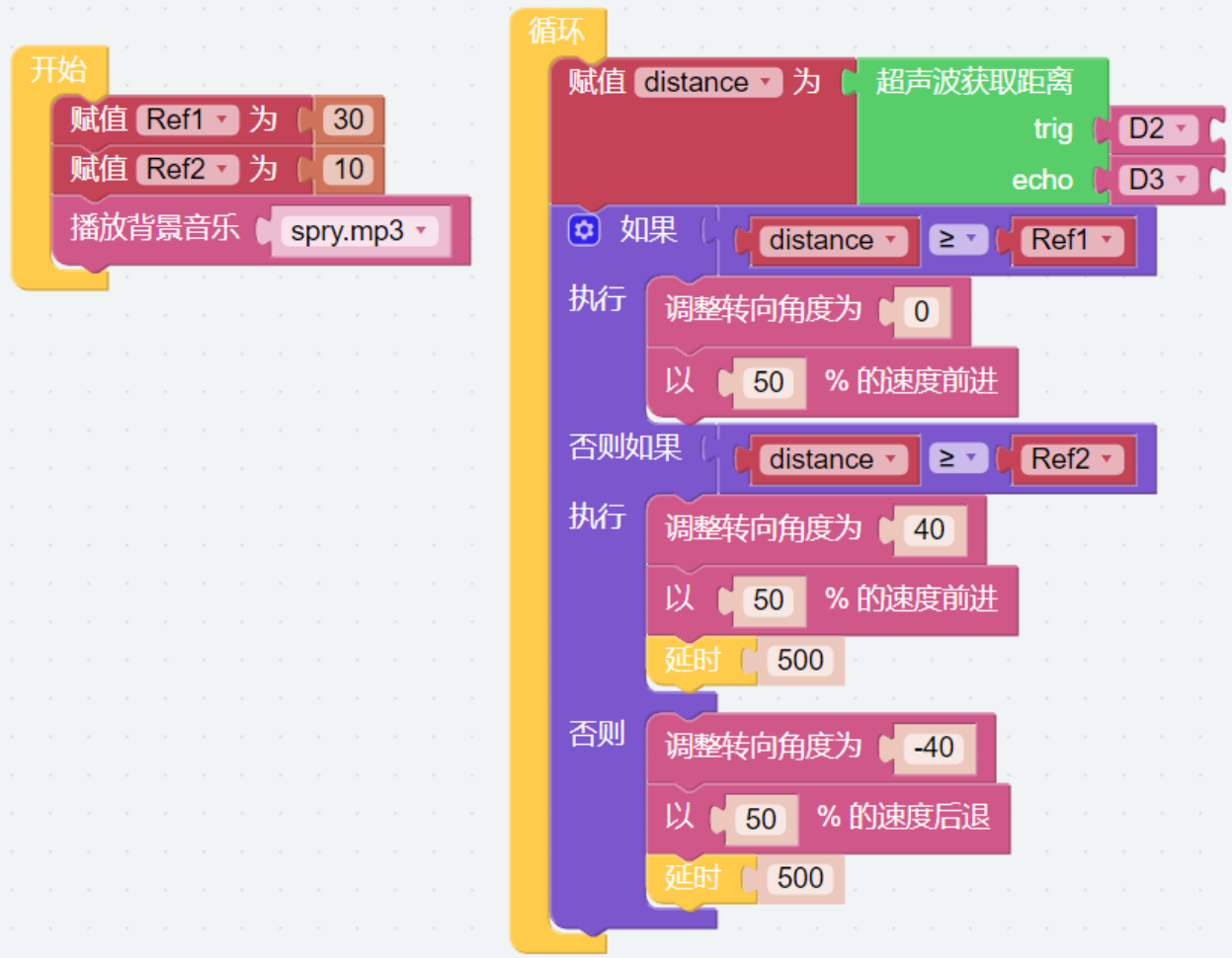
提示



要实现多个条件判断，则要将简单的 **如果-执行-否则**块更改为 **如果-执行-否则如果-执行-否则**块。这是通过单击如上所示的设置图标来完成的。

示例

备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.



代码运行后，小车将边播放音乐边前进；若碰到障碍物，小车将转向其他方向继续前进。

4.15 悬崖检测

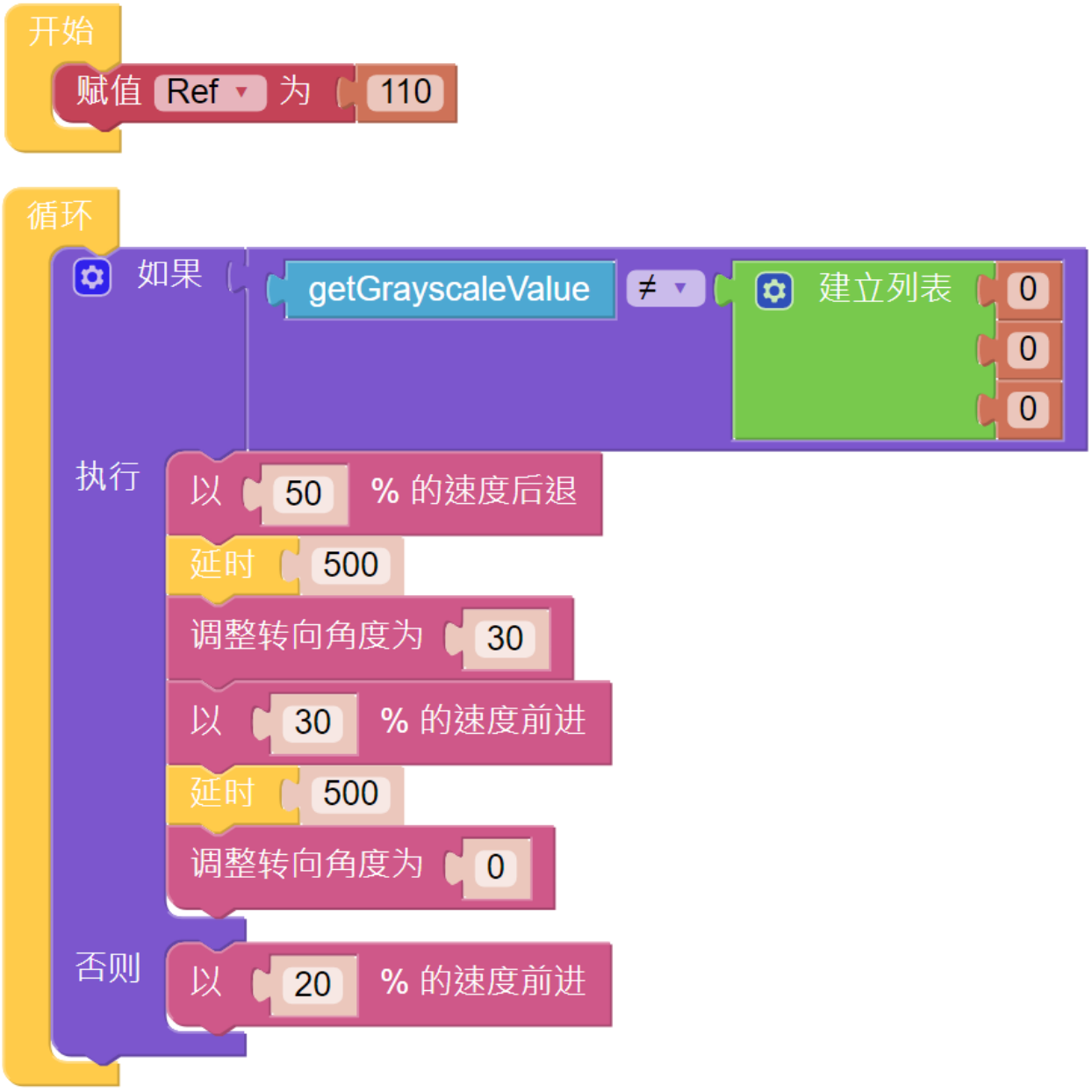
该项目将使用 **灰度模块**来防止帕克在自由移动时从悬崖上掉下来。

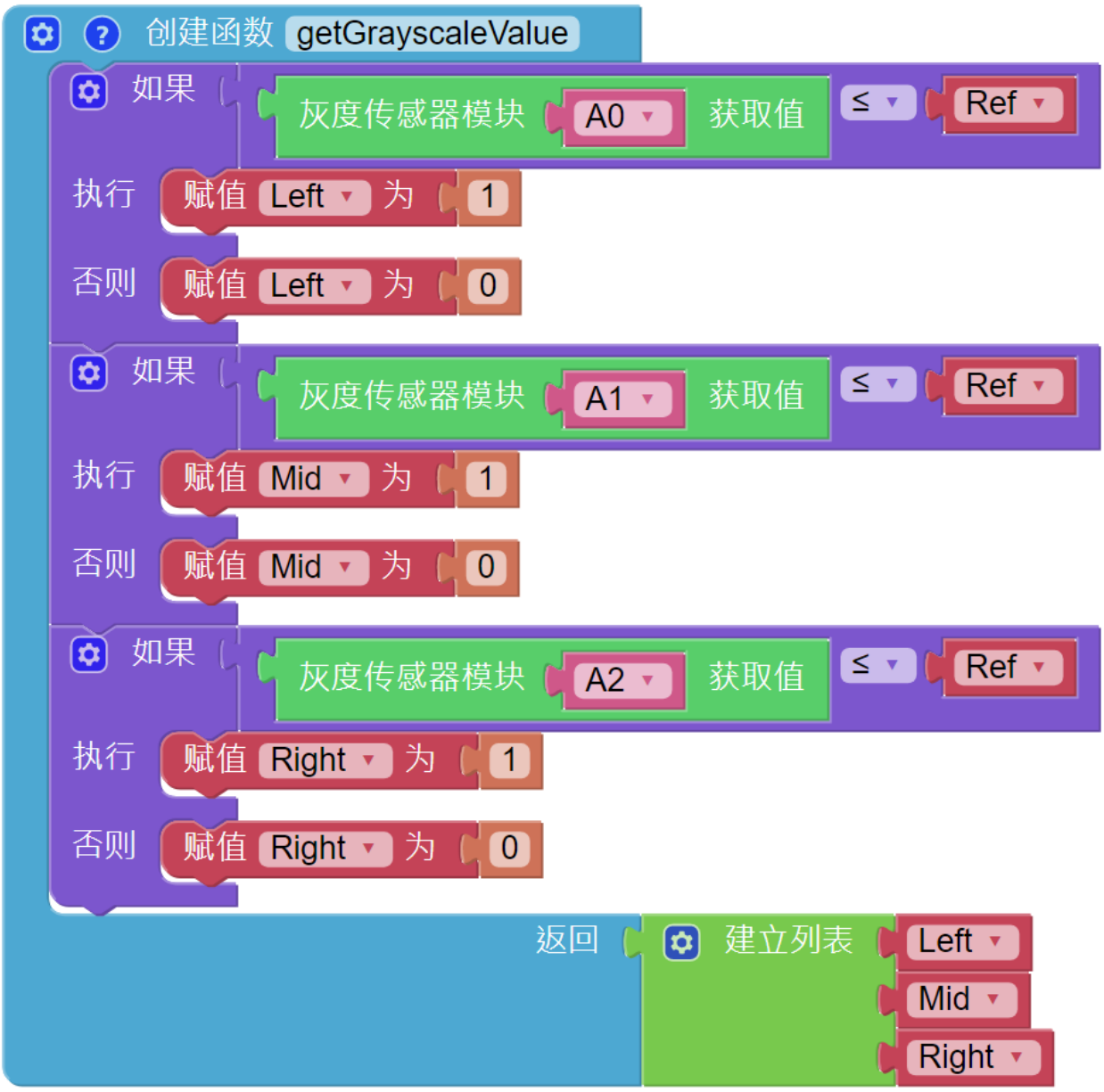
提示

灰度模块将多次执行相同的操作。为了简化程序，该项目引入了一个 **创建函数**，它将向 **循环**块返回一个 **列表**变量。

例子

备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.


开始
赋值 Ref 为 110
循环
如果 getGrayscaleValue ≠ 建立列表 0 0 0
执行
以 50 % 的速度后退
延时 500
调整转向角度为 30
以 30 % 的速度前进
延时 500
调整转向角度为 0
否则
以 20 % 的速度前进



代码运行后，小车会向前移动，当遇到悬崖，它将后退并向其他方向前进。

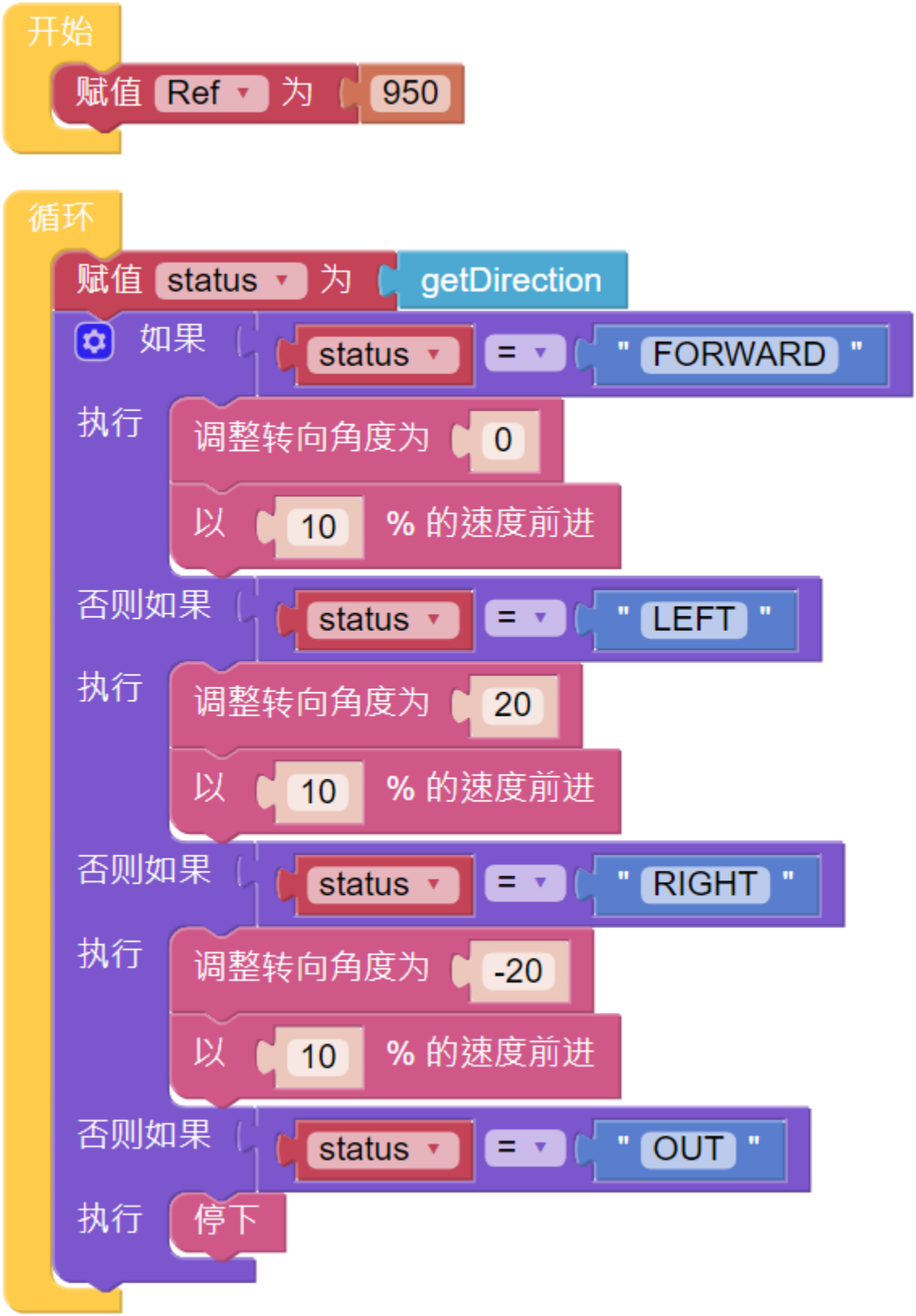
4.16 矿车

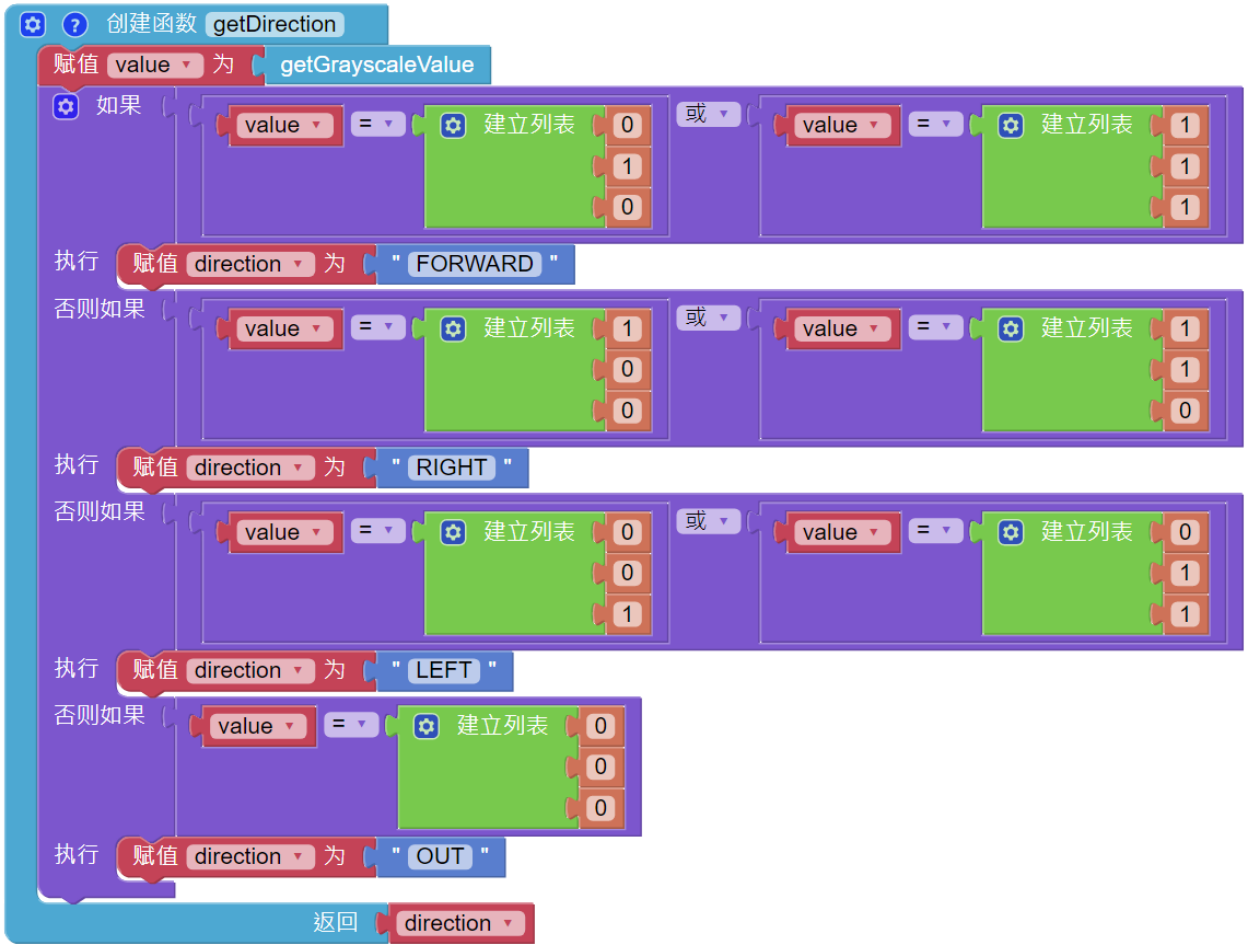
让我们做一个矿车项目吧！本项目将使用灰度模块使帕克沿着轨道向前移动。使用深色胶带在地面上制作一条轨道，不要有太大的转弯幅度，否则可能脱离轨道。

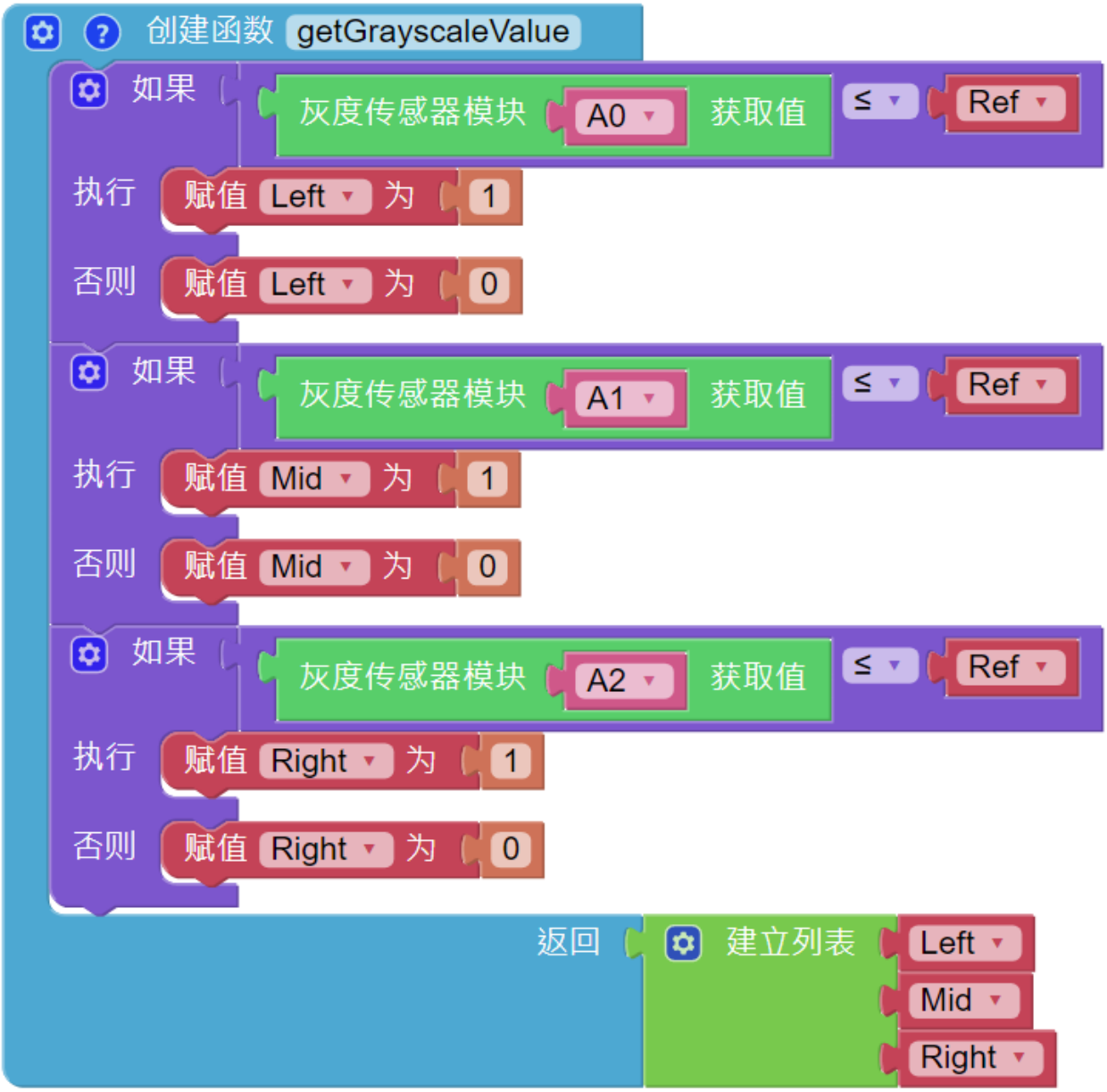
沿着轨道移动时，灰度模块左右两侧的探头会检测到浅色地面，中间的探头会检测到轨道。如果轨道有弧线，传感器左侧或右侧的探头将检测到深色胶带，并朝该方向转动车轮。如果矿车到达轨道末端或脱轨，灰度模块将不再检测到深色磁带轨道，帕克将停下来。

例子

备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.

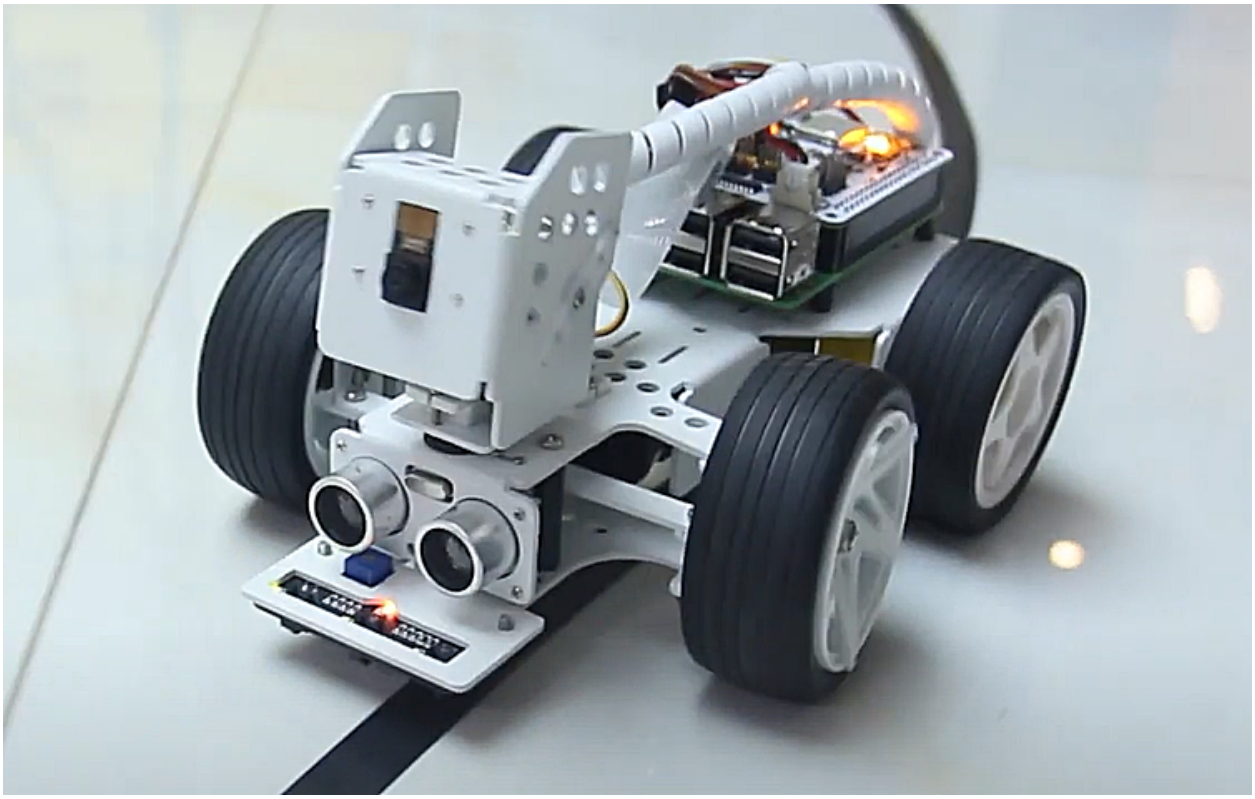

开始
赋值 Ref 为 950
循环
赋值 status 为 getDirection
如果 status = " FORWARD "
执行
调整转向角度为 0
以 10 % 的速度前进
否则如果 status = " LEFT "
执行
调整转向角度为 20
以 10 % 的速度前进
否则如果 status = " RIGHT "
执行
调整转向角度为 -20
以 10 % 的速度前进
否则如果 status = " OUT "
执行
停下


创建函数 getDirection
赋值 value 为 getGrayscaleValue
如果 value = 建立列表 0 1 0 或 value = 建立列表 1 1 1
执行 赋值 direction 为 " FORWARD "
否则如果 value = 建立列表 1 0 0 或 value = 建立列表 1 1 0
执行 赋值 direction 为 " RIGHT "
否则如果 value = 建立列表 0 0 1 或 value = 建立列表 0 1 1
执行 赋值 direction 为 " LEFT "
否则如果 value = 建立列表 0 0 0
执行 赋值 direction 为 " OUT "
返回 direction



4.17 矿车 +

在这个项目是在《矿车》项目中添加了脱轨恢复功能，让帕克适应更弯曲的轨道。



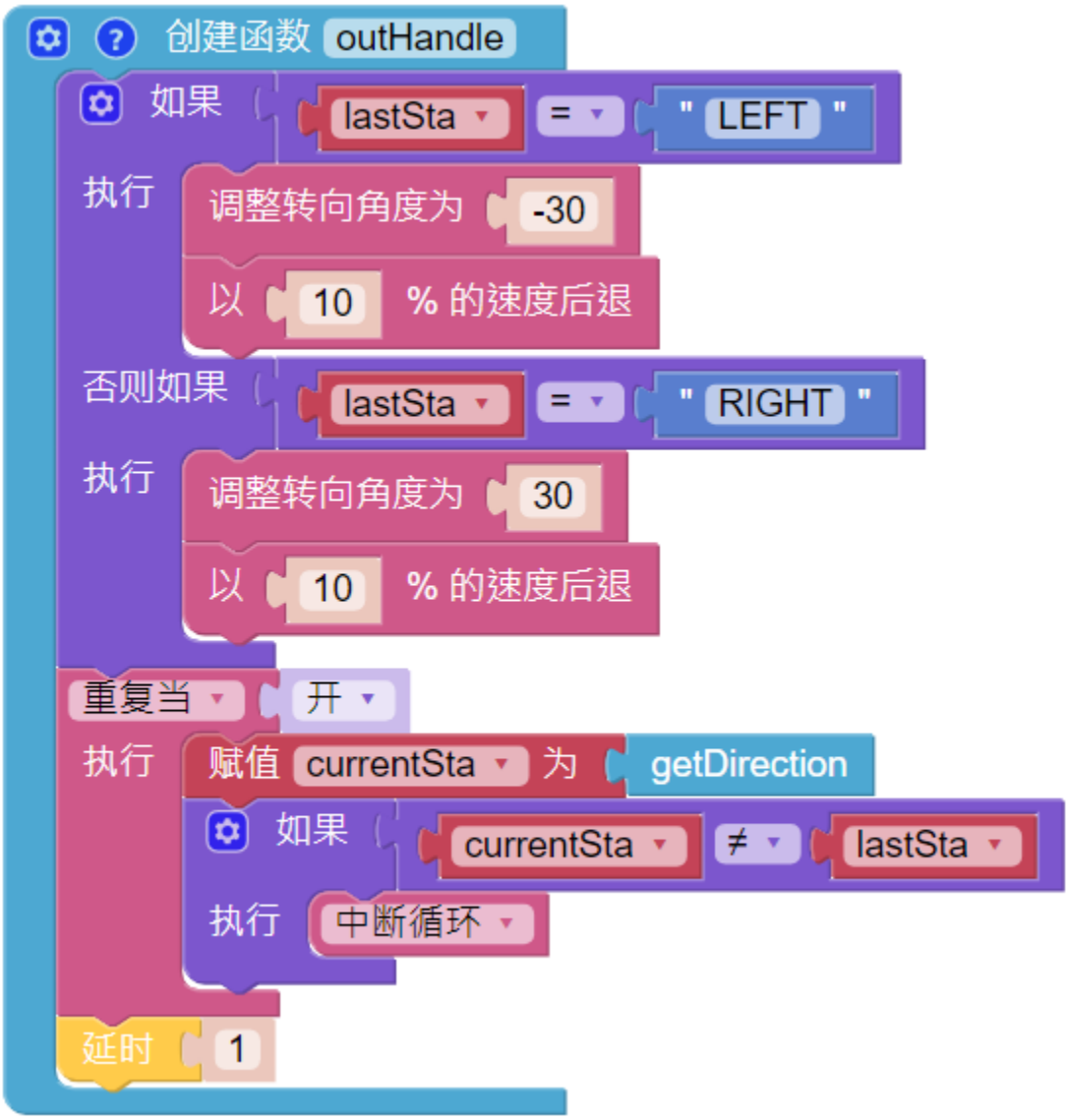
提示

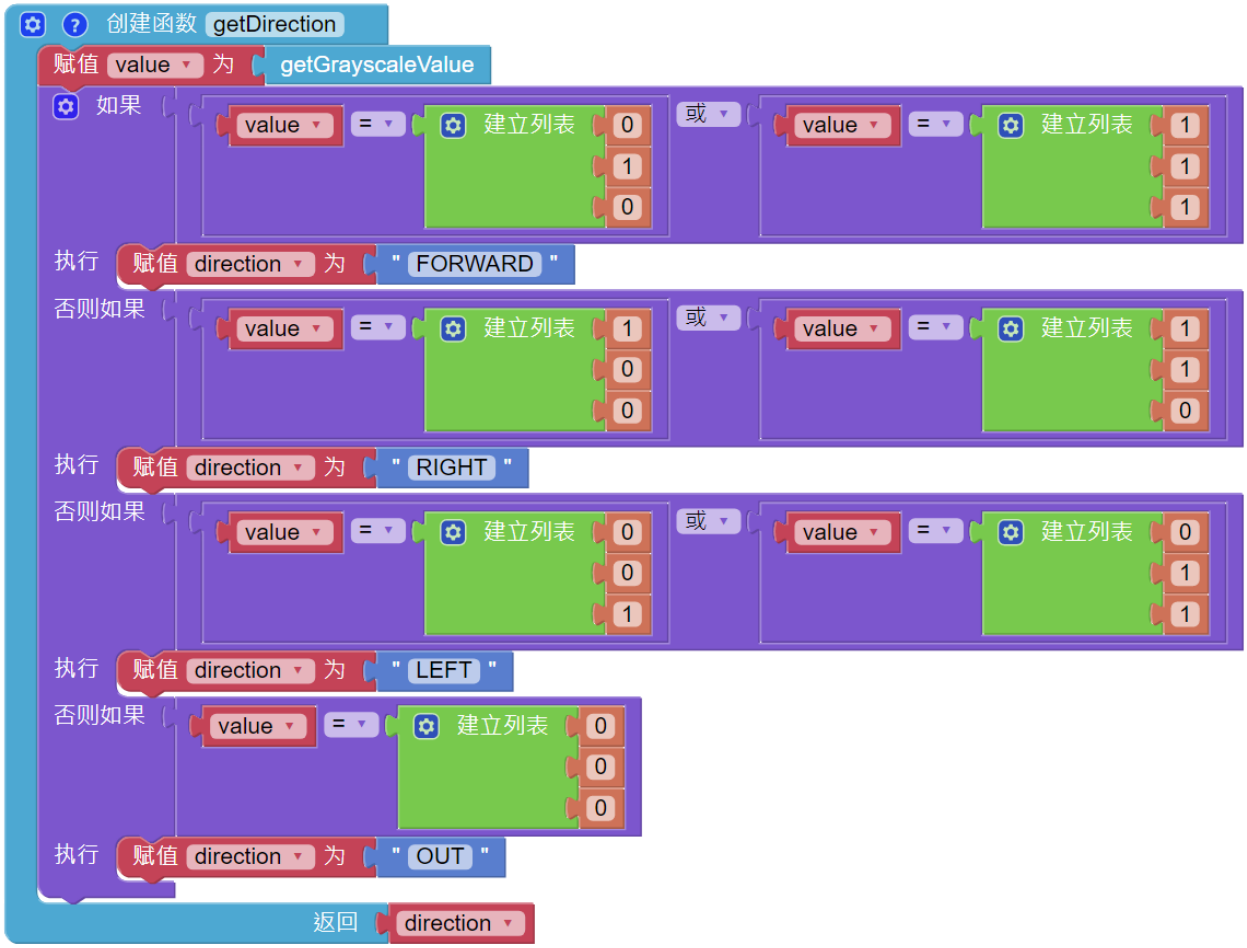
使用另一个 **创建函数**块让帕克备份并从急转弯中恢复。请注意，新的 **创建函数**函数不返回任何值，而仅用于重新定向帕克。

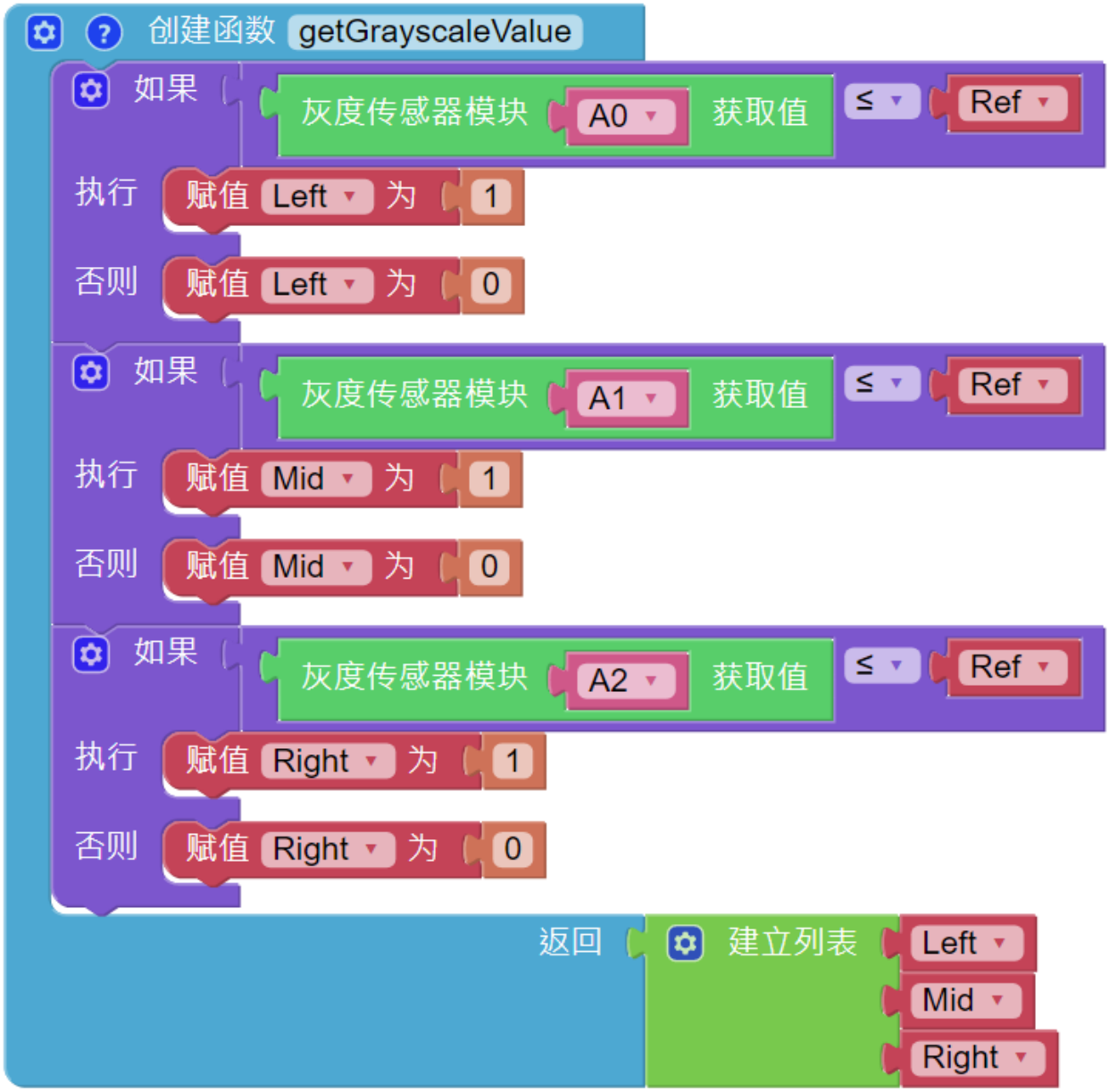
示例

备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.

开始
赋值 Ref 为 950
循环
赋值 sta 为 getDirection
如果 sta ≠ " OUT "
执行 赋值 lastSta 为 sta
如果 sta = " FORWERD "
执行 调整转向角度为 0
以 10 % 的速度前进
否则如果 sta = " LEFT "
执行 调整转向角度为 20
以 10 % 的速度前进
否则如果 sta = " RIGHT "
执行 调整转向角度为 -20
以 10 % 的速度前进
否则如果 sta = " OUT "
执行 outHandle


创建函数 outHandle
如果 lastSta = " LEFT "
执行
调整转向角度为 -30
以 10 % 的速度后退
否则如果 lastSta = " RIGHT "
执行
调整转向角度为 30
以 10 % 的速度后退
重复当 开
执行
赋值 currentSta 为 getDirection
如果 currentSta ≠ lastSta
执行 中断循环
延时 1


创建函数 getDirection
赋值 value 为 getGrayscaleValue
如果
value = 建立列表 0 1 0
或
value = 建立列表 1 1 1
执行 赋值 direction 为 " FORWARD "
否则如果
value = 建立列表 1 0 0
或
value = 建立列表 1 1 0
执行 赋值 direction 为 " RIGHT "
否则如果
value = 建立列表 0 0 1
或
value = 建立列表 0 1 1
执行 赋值 direction 为 " LEFT "
否则如果
value = 建立列表 0 0 0
执行 赋值 direction 为 " OUT "
返回 direction



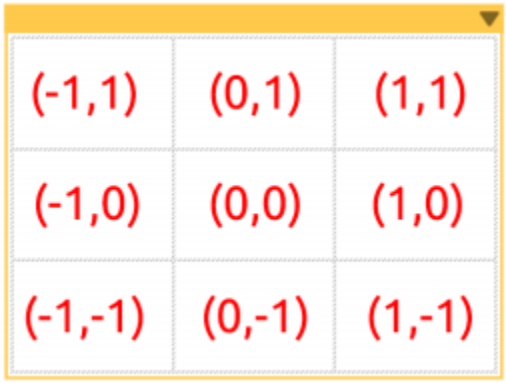
4.18 斗牛

将帕克变成愤怒的公牛！准备一块红布，如手帕，成为一名斗牛士。帕克追红布的时候注意不要被撞到！

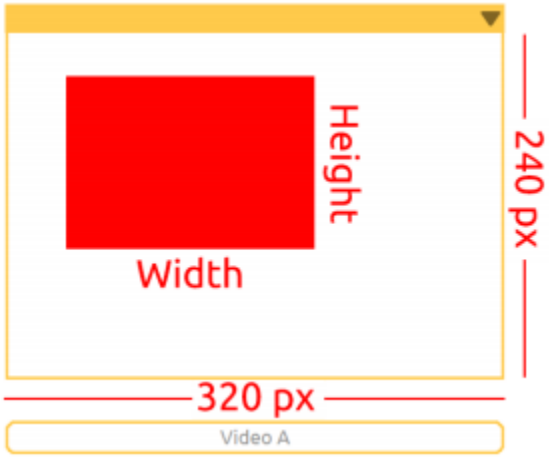
备注： 这个项目比前面的项目更进阶。帕克需要使用颜色检测功能让相机保持朝向红布，然后身体方向需要根据相机朝向的方向自动调整。

提示

首先将 **颜色检测 [红色]** 块添加到 **开始**小部件，以使帕克寻找红色对象。在永久循环中，添加 **颜色检测的 [宽度]** 块以将输入转换为 对象检测网格。



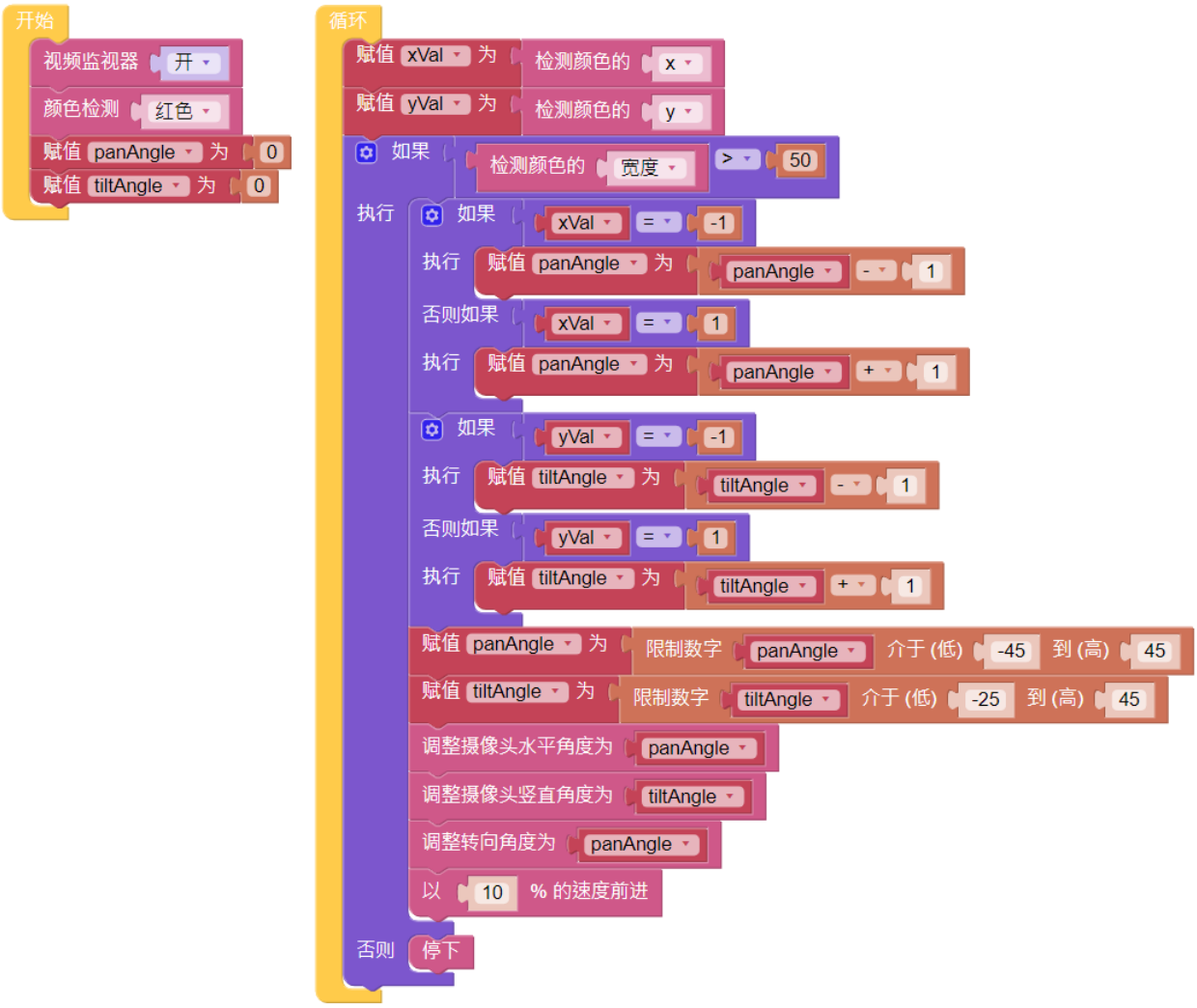
对象检测将以 (x, y) 值输出检测到的坐标，基于相机图像的中心点。屏幕被分成 3x3 的网格，如下图，因此，如果红布保持在相机图像的左上角，则 (x, y) 坐标将为 (-1, 1)。



对象检测将检测图形的宽度和高度。如果识别出多个目标，则记录最大目标的尺寸。

示例

备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.



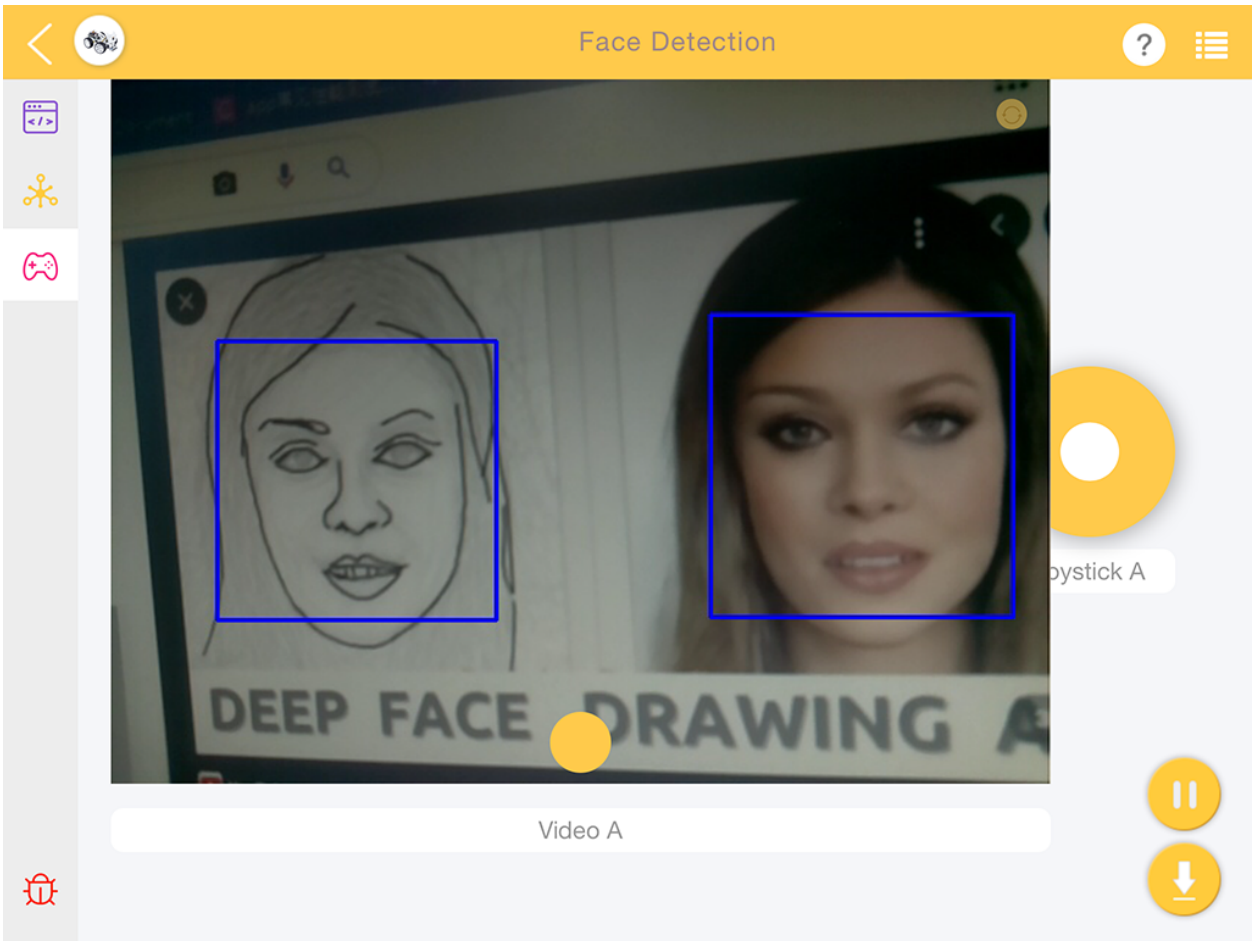
代码运行后，你就可以用一块红布或红球来让小车跟着走。

4.19 礼让行人

该项目将使帕克根据路况采取适当的措施。在行驶过程中，如果在其路径上检测到行人，帕克将完全停止。

程序运行后，在帕克前面放一张人的照片。视频监视器将检测到人脸，帕克将自动停止。

为了模拟驾驶安全协议，创建了一个判断程序，该程序将向 **如果-执行-否则**块发送 **[count]** 值。判断程序会寻找人脸 10 次，如果确实出现了人脸，则将 **[count]** 增加 +1。当 **[count]** 大于 3 时，帕克将停止移动。



示例

备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.

程序运行后，在帕克前面放一张人的照片。视频监视器将检测到人脸，帕克将自动停止。

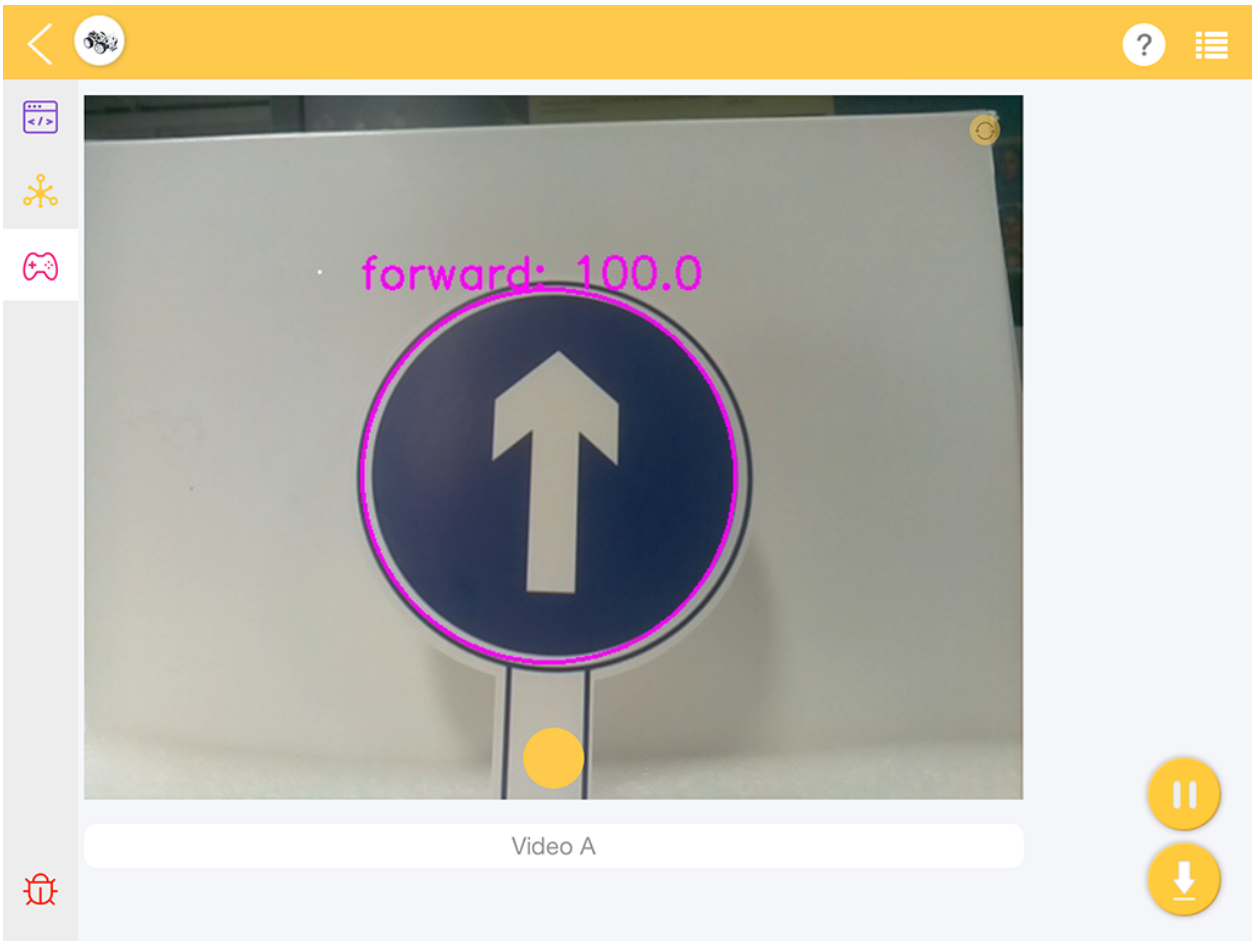
4.20 交通标志检测

该项目使用图像检测功能来寻找交通标志，并使帕克按照标志上的说明进行操作。**交通标志检测**块启动后，将识别以下 PDF 中包含的 4 种不同交通标志模型。当帕克检测到 **停止（STOP）**标志时，它会停下来，**向前（FORWARD）**标志将使其向前行驶，而 **向左（LEFT）**或 **向右（RIGHT）**箭头将使其转向该方向。

- [PDF] 交通标志卡

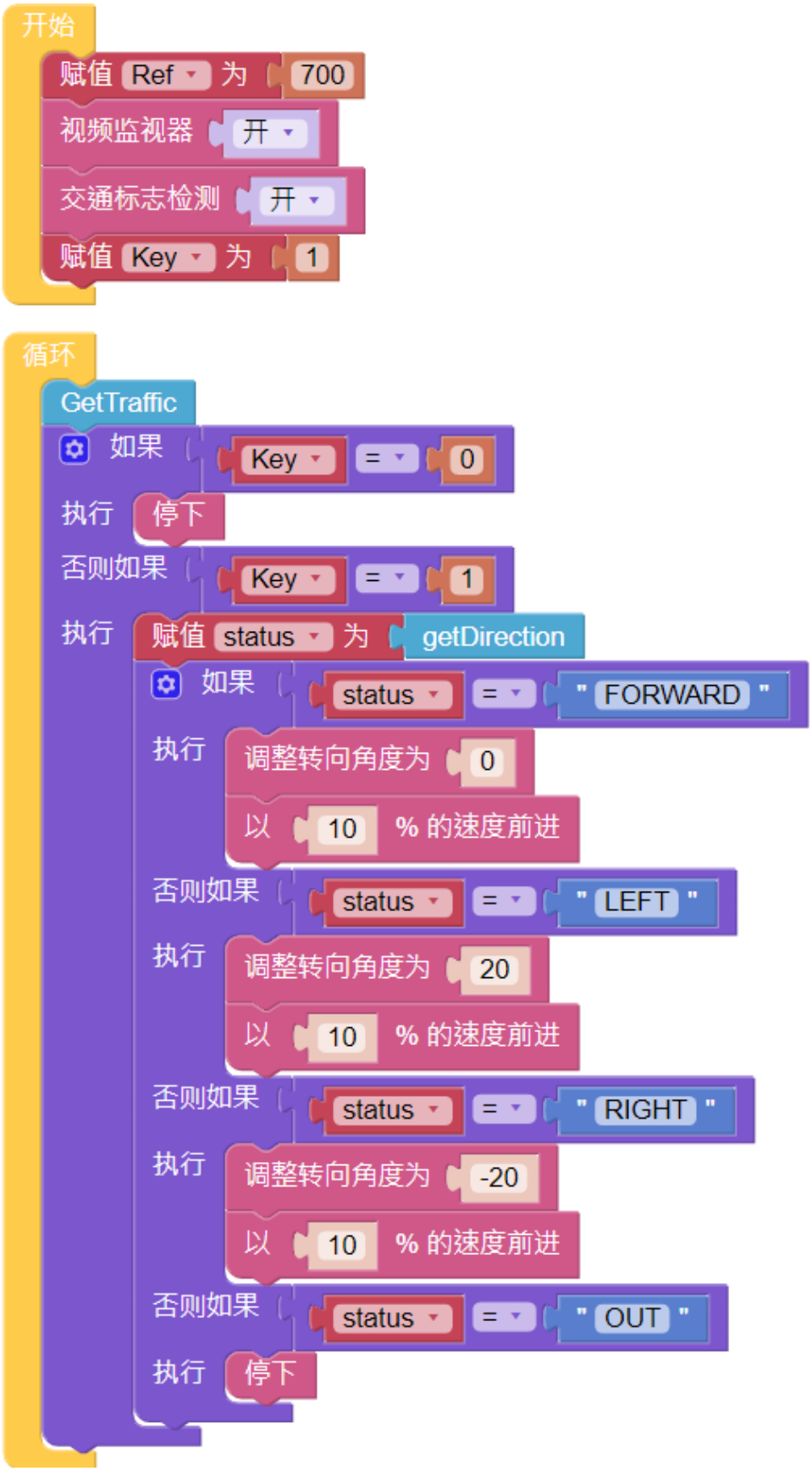
备注： 由于打印机碳粉或打印介质（如棕褐色纸）的不同，打印的交通标志颜色可能与 EzBlock 颜色模型的色调略有不同。这会导致不太准确的颜色识别。

该项目基于*矿车* ，但帕克使用一种算法进行交通标志检测，而不是使用灰度传感器。检测结果可以通过 EzBlock Studio 中的视频监视器查看。



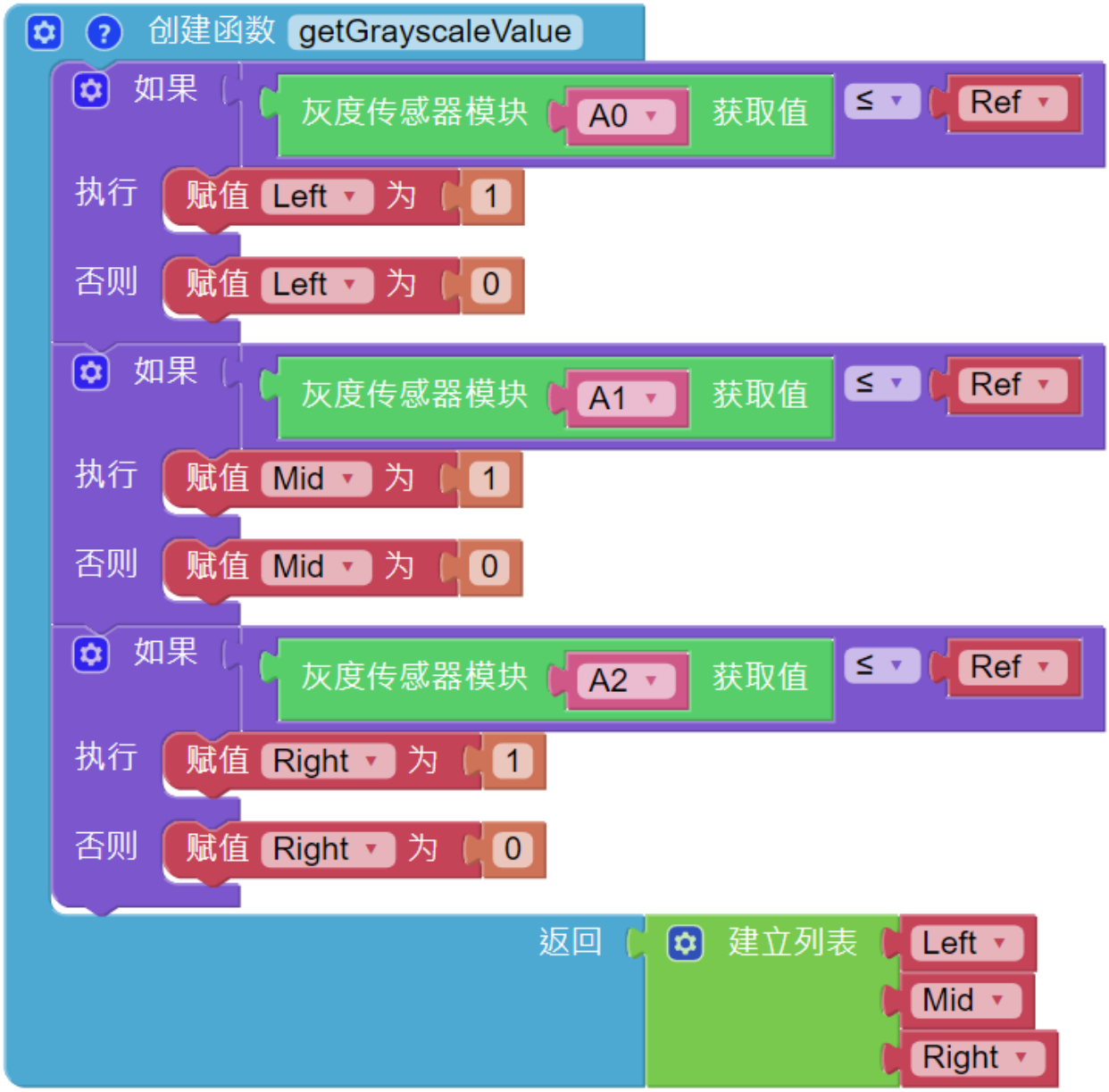
示例

备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.


开始
赋值 Ref 为 700
视频监视器 开
交通标志检测 开
赋值 Key 为 1
循环
GetTraffic
如果 Key = 0
执行 停下
否则如果 Key = 1
执行 赋值 status 为 getDirection
如果 status = " FORWARD "
执行 调整转向角度为 0
以 10 % 的速度前进
否则如果 status = " LEFT "
执行 调整转向角度为 20
以 10 % 的速度前进
否则如果 status = " RIGHT "
执行 调整转向角度为 -20
以 10 % 的速度前进
否则如果 status = " OUT "
执行 停下

创建函数 GetTraffic
赋值 Traffic 为 被交通标志的 类型
打印 Traffic
如果 Traffic = " stop "
执行 赋值 Key 为 0
否则如果 Traffic = " forward "
执行 赋值 Key 为 1

创建函数 getDirection
赋值 value 为 getGrayscaleValue
如果 value = 建立列表 0 1 0 或 value = 建立列表 1 1 1
执行 赋值 direction 为 " FORWARD "
否则如果 value = 建立列表 1 0 0 或 value = 建立列表 1 1 0
执行 赋值 direction 为 " RIGHT "
否则如果 value = 建立列表 0 0 1 或 value = 建立列表 0 1 1
执行 赋值 direction 为 " LEFT "
否则如果 value = 建立列表 0 0 0
执行 赋值 direction 为 " OUT "
返回 direction



4.21 定向越野

本项目使用遥控功能引导帕克进行竞技寻宝游戏!

首先，设置障碍训练场、迷宫，甚至是帕克可以通过的空房间。然后，沿路线随机放置六个标记，并在六个标记中的每个标记处放一张色卡，供帕克查找。

帕克的六种颜色模型是：红色、橙色、黄色、绿色、蓝色和紫色，可以从下面的 PDF 中的彩色打印机进行打印。

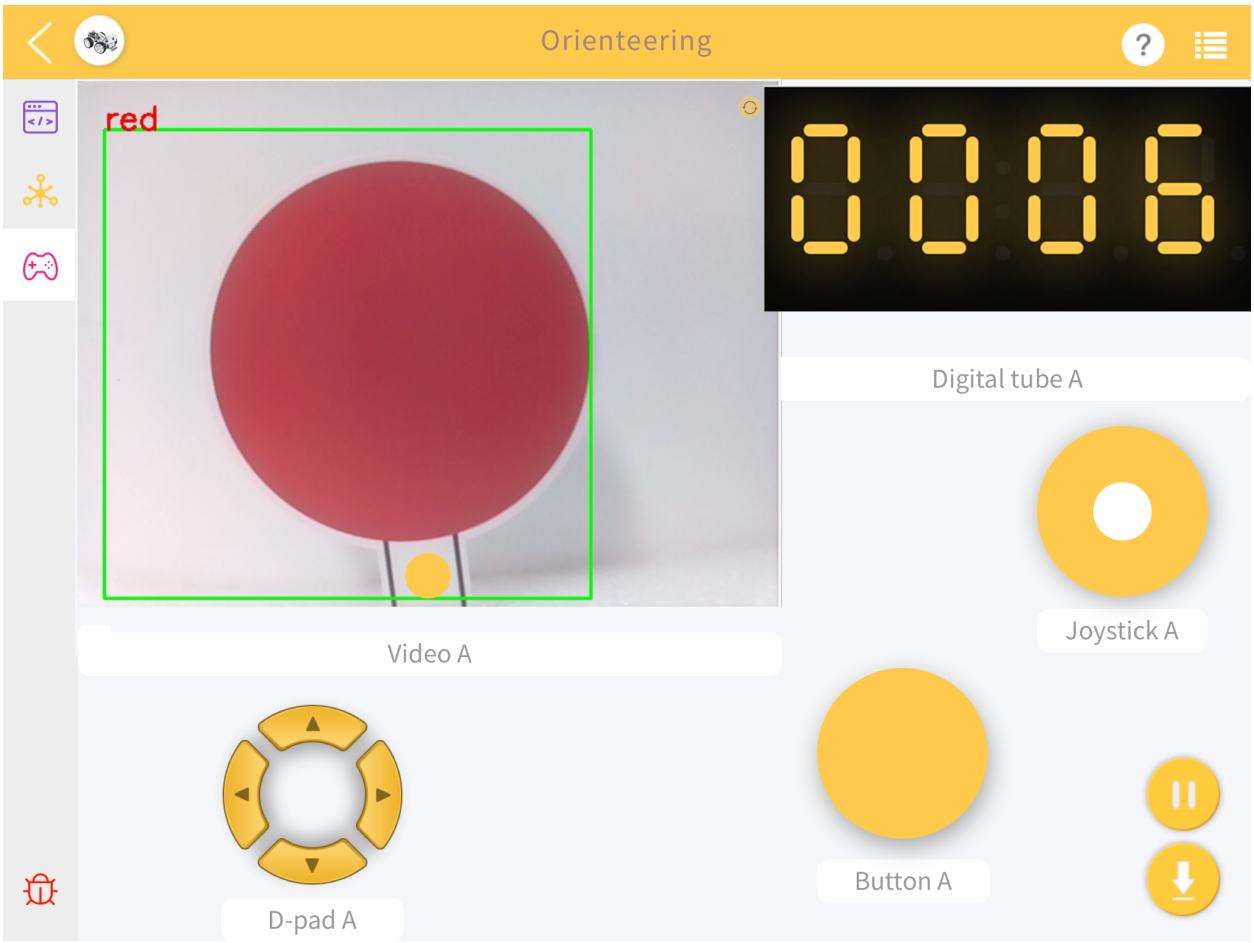
- [PDF] 颜色卡

备注： 由于打印机碳粉或打印介质（如棕褐色纸）的不同，打印的颜色可能与 Ezblock 颜色模型的色调略有不同。这会导致不太准确的颜色识别。

帕克将被编程为以随机顺序查找六种颜色中的三种，并将使用 TTS 功能宣布下一个要查找的颜色。

目标是帮助帕克在尽可能短的时间内找到三种颜色中的每一种。

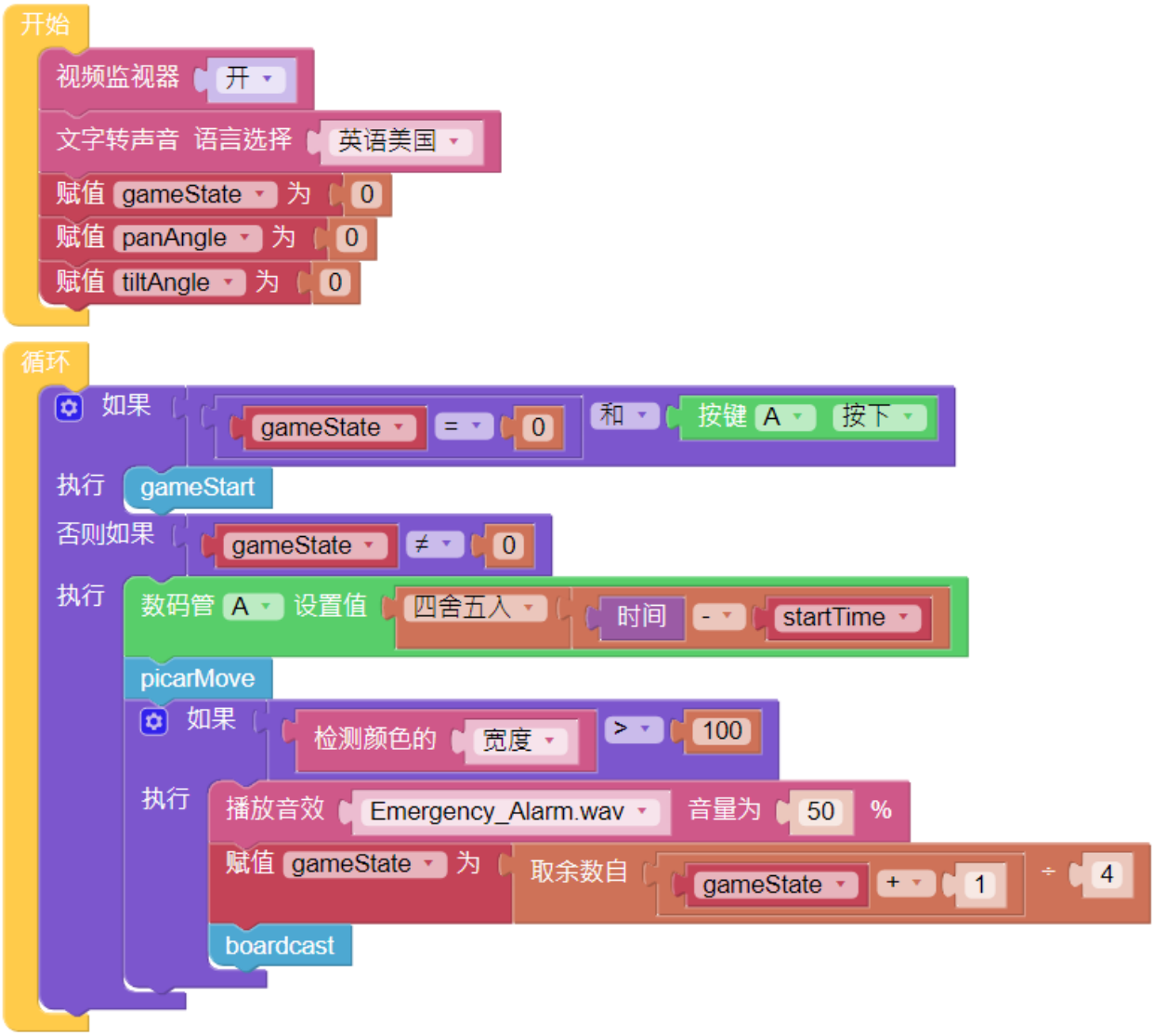
将帕克放置在场地中间，点击遥控页面上的按钮开始游戏。

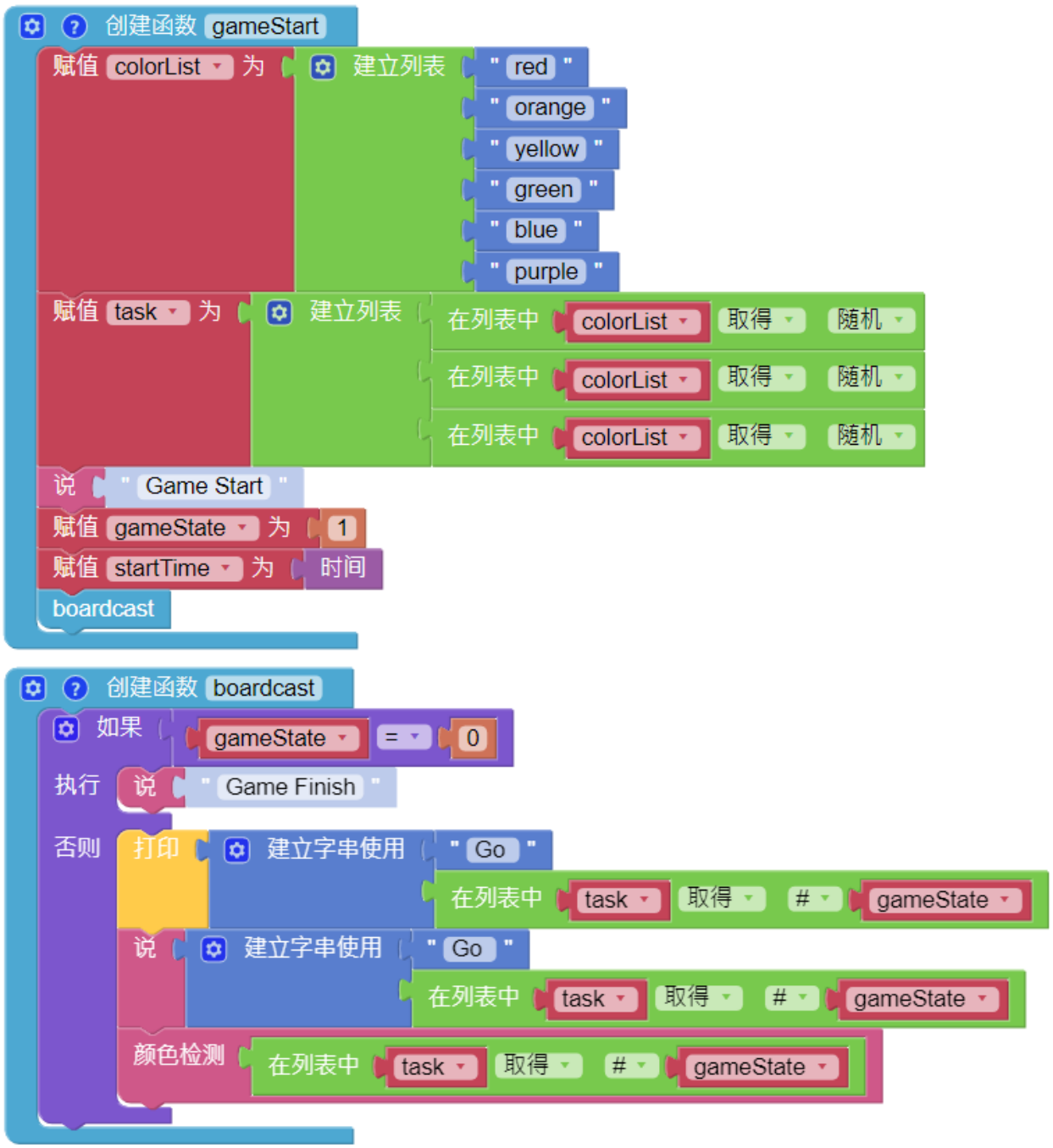


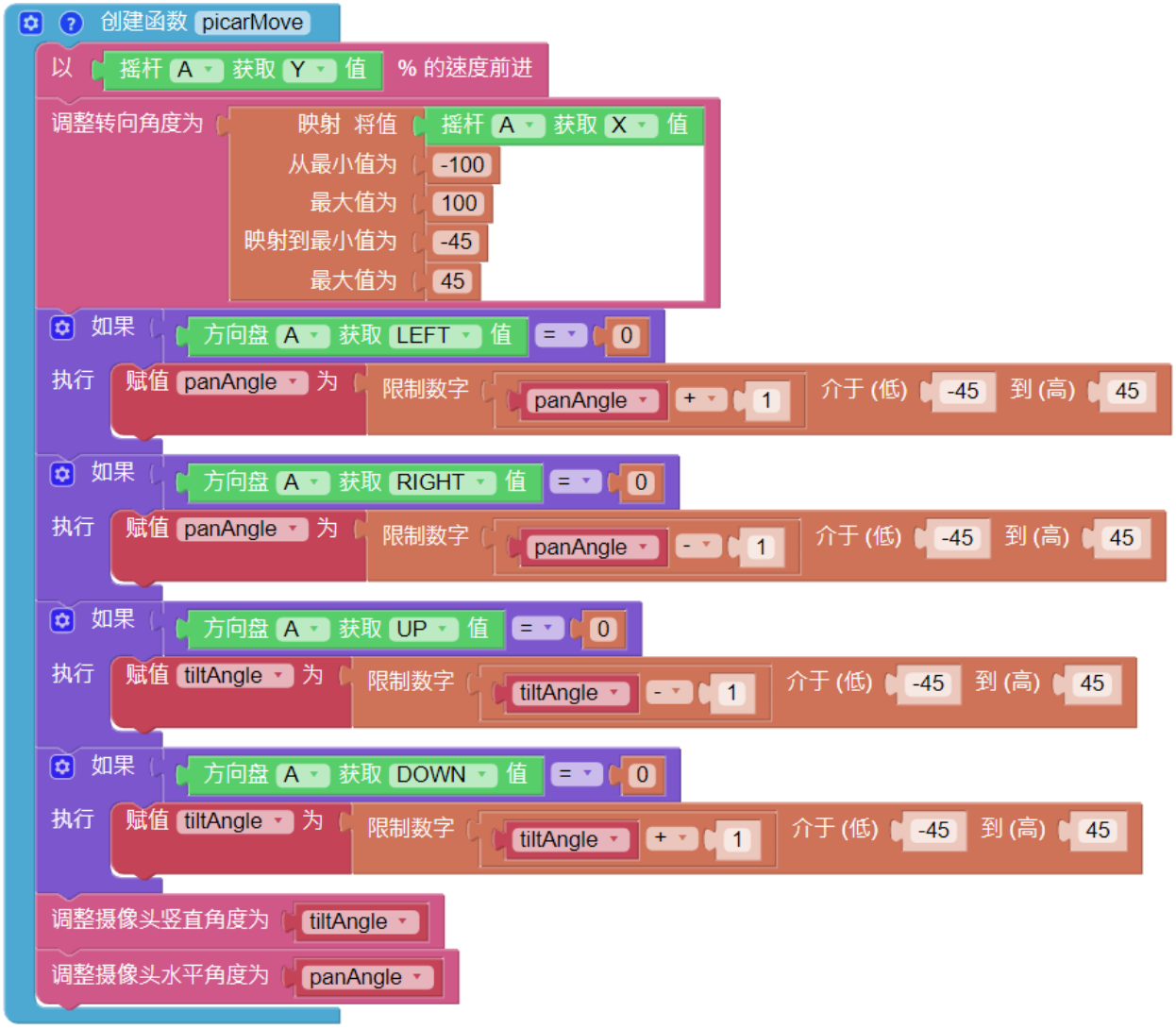
和朋友轮流玩这个游戏，看看谁能最快帮助帕克完成目标！

示例

备注： 你可以直接打开我们提供的示例或者是按照下图来编写程序，详细教程请参考打开或新建项目.


开始
视频监视器 开
文字转声音 语言选择 英语美国
赋值 gameState 为 0
赋值 panAngle 为 0
赋值 tiltAngle 为 0
循环
如果 gameState = 0 和 按键 A 按下
执行 gameStart
否则如果 gameState ≠ 0
执行 数码管 A 设置值 四舍五入 时间 - startTime
picarMove
如果 检测颜色的 宽度 > 100
执行 播放音效 Emergency_Alarm.wav 音量为 50 %
赋值 gameState 为 取余数自 gameState + 1 ÷ 4
boardcast


创建函数 gameStart
赋值 colorList 为 建立列表 " red " " orange " " yellow " " green " " blue " " purple "
赋值 task 为 建立列表 在列表中 colorList 取得 随机 在列表中 colorList 取得 随机 在列表中 colorList 取得 随机
说 " Game Start "
赋值 gameState 为 1
赋值 startTime 为 时间
boardcast
创建函数 boardcast
如果 gameState = 0
执行 说 " Game Finish "
否则 打印 建立字串使用 " Go " 在列表中 task 取得 # gameState
说 建立字串使用 " Go " 在列表中 task 取得 # gameState
颜色检测 在列表中 task 取得 # gameState


创建函数 picarMove
以 摇杆 A 获取 Y 值 % 的速度前进
调整转向角度为 映射 将值 摇杆 A 获取 X 值
从最小值为 -100
最大值为 100
映射到最小值为 -45
最大值为 45
如果 方向盘 A 获取 LEFT 值 = 0
执行 赋值 panAngle 为 限制数字 panAngle + 1 介于 (低) -45 到 (高) 45
如果 方向盘 A 获取 RIGHT 值 = 0
执行 赋值 panAngle 为 限制数字 panAngle - 1 介于 (低) -45 到 (高) 45
如果 方向盘 A 获取 UP 值 = 0
执行 赋值 tiltAngle 为 限制数字 tiltAngle - 1 介于 (低) -45 到 (高) 45
如果 方向盘 A 获取 DOWN 值 = 0
执行 赋值 tiltAngle 为 限制数字 tiltAngle + 1 介于 (低) -45 到 (高) 45
调整摄像头竖直角度为 tiltAngle
调整摄像头水平角度为 panAngle

CHAPTER 5

玩转 Python

对于希望用 Python 编程的新手和初学者，需要一些基本的 Python 编程技巧和树莓派操作系统的知识。要开始配置树莓派，请参考 Python 快速指南。

5.1 Python 快速指南

本节教大家如何安装树莓派操作系统，配置树莓派 wifi，远程访问树莓派运行相应代码。

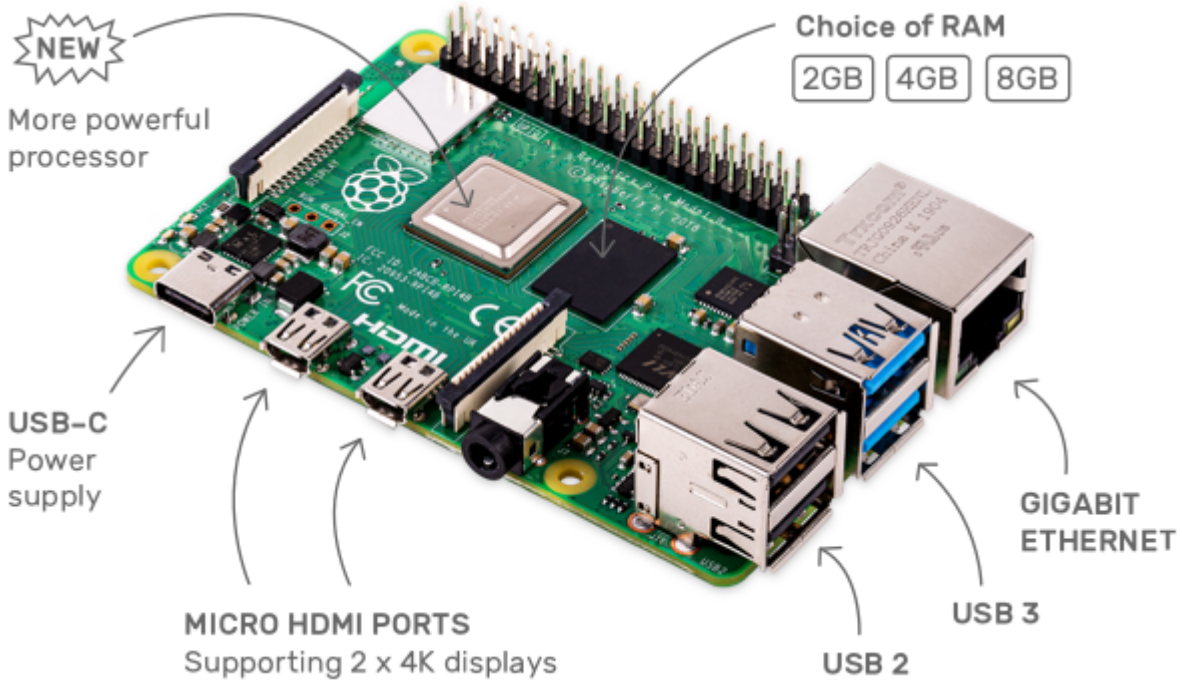
如果您熟悉树莓派并且可以成功打开命令行，那么您可以跳过前 3 部分，然后完成后面两部分。

5.1.1 我们需要什么？

所需组件

树莓派

树莓派是一款低成本、信用卡大小的计算机，可插入进入电脑显示器或电视，并使用标准键盘和鼠标。这是一款功能强大的小型设备，可让所有年龄段的人探索计算，并学习如何使用 Scratch 和 Python 等语言进行编程。



电源适配器

为了连接到电源插座，树莓派有一个微型 USB 端口（在许多手机上都可以找到）。您将需要一个提供至少 2.5 安培电流的电源。

微型 SD 卡

您的树莓派需要一张 Micro SD 卡来存储其所有文件和树莓派操作系统。您需要一张容量至少为 8 GB 的微型 SD 卡。

可选组件

屏幕

要查看树莓派的桌面环境，需要使用的屏幕可以是电视屏幕，也可以是电脑显示器。如果屏幕有内置扬声器，Pi 通过它们播放声音。

鼠标和键盘

使用屏幕时，还需要一个 USB 键盘和一个 USB 鼠标。

HDMI

树莓派有一个 HDMI 输出端口，与大多数现代电视和计算机显示器的 HDMI 端口兼容。如果您的屏幕只有 DVI 或 VGA 端口，则需要使用合适的转换线。

外壳

你可以把树莓派放在一个盒子里；通过这种方式，您可以保护您的设备。

声音或耳机

树莓派配备了一个 3.5mm 左右的音频接口，可以在你的屏幕没有内置扬声器或者没有屏幕操作的情况下使用。

5.1.2 安装操作系统

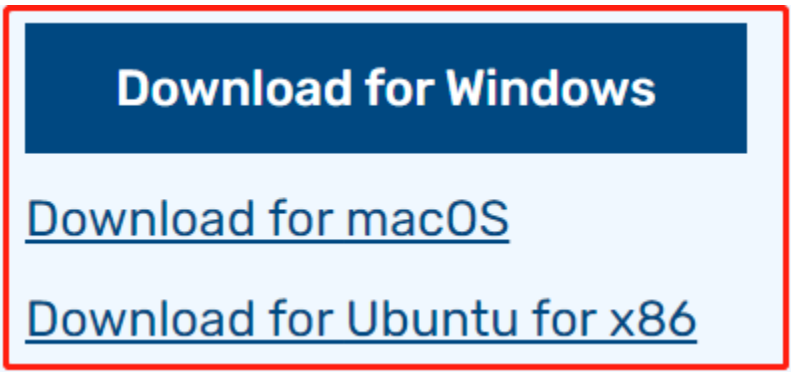
必需组件

任意树莓派	1 * 个人计算机
1 * 微型 SD 卡	

第 1 步

树莓派开发了一个图形 SD 卡写入工具，适用于 Mac OS、Ubuntu 18.04 和 Windows，对于大多数用户来说是最简单的选择，因为它会下载映像并将其自动安装到 SD 卡。

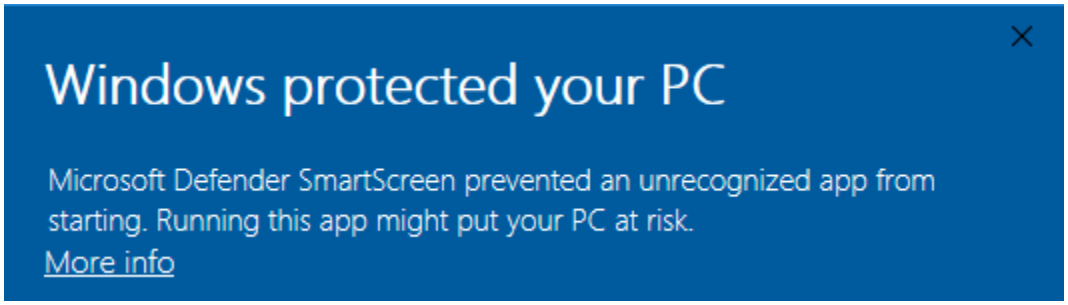
访问下载页面：https://www.raspberrypi.org/software/。单击与您的操作系统匹配的 Raspberry Pi Imager 链接，下载完成后，单击它以启动安装程序。



第 2 步

当您启动安装程序时，您的操作系统可能会尝试阻止您运行它。例如，在 Windows 上，我收到以下消息：

如果出现此消息，请点击 **更多信息**，然后点击 **仍然运行**，然后按照说明安装 Raspberry Pi Imager。



第 3 步

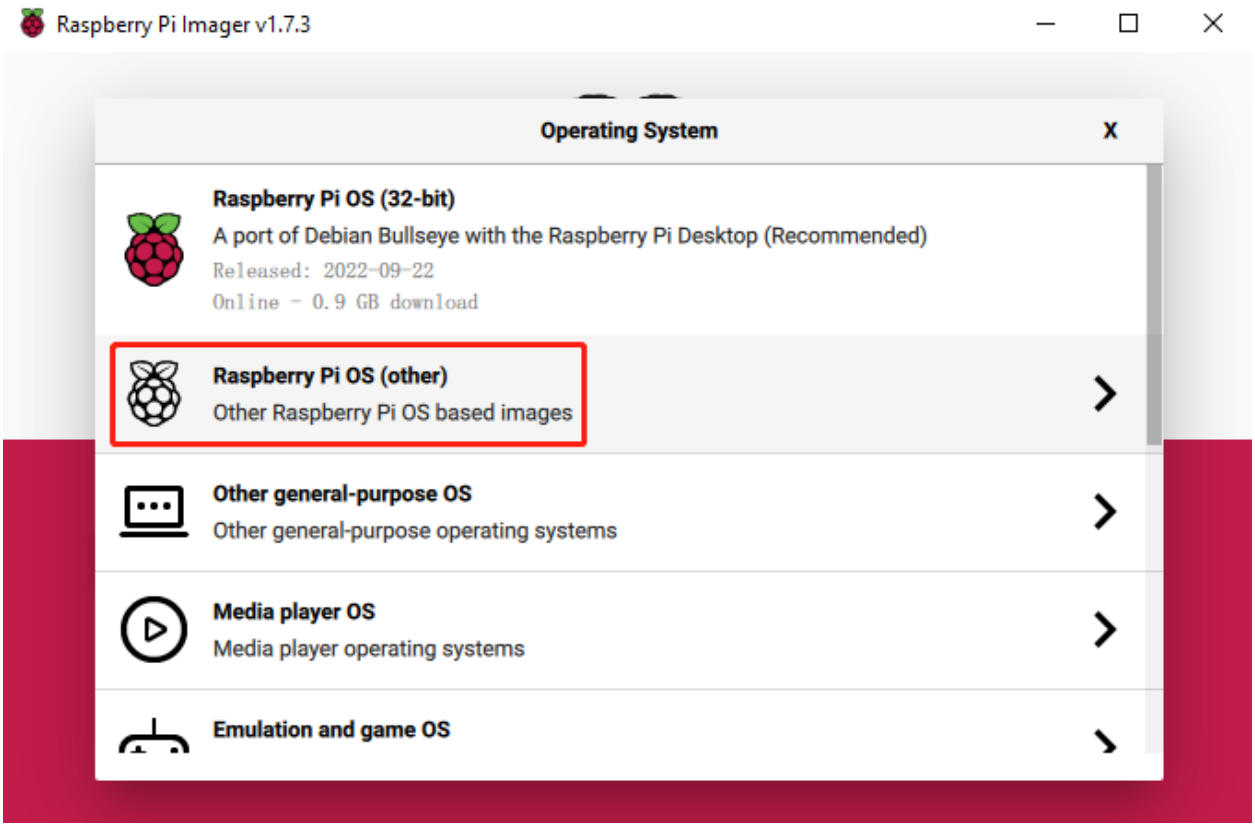
将 SD 卡插入计算机或笔记本电脑的 SD 卡插槽。

第 4 步

在 Raspberry Pi Imager 中，单击选择 **操作系统 -> Raspberry Pi OS(Legacy)**。

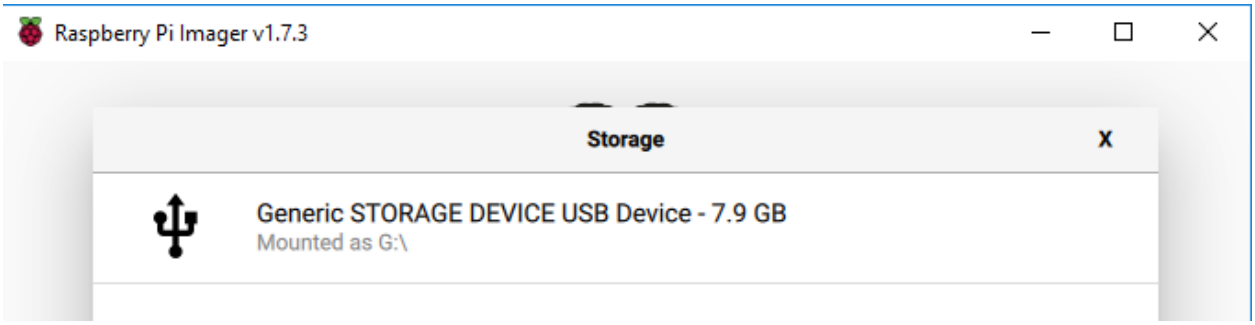
警告:

- 请不要安装 **Bookworm** 版本，因为扬声器将无法工作。
- 您需要安装 **Raspberry Pi OS（legacy）** 版本 - **Debian Bullseye**。



第 5 步

选择您正在使用的 SD 卡。

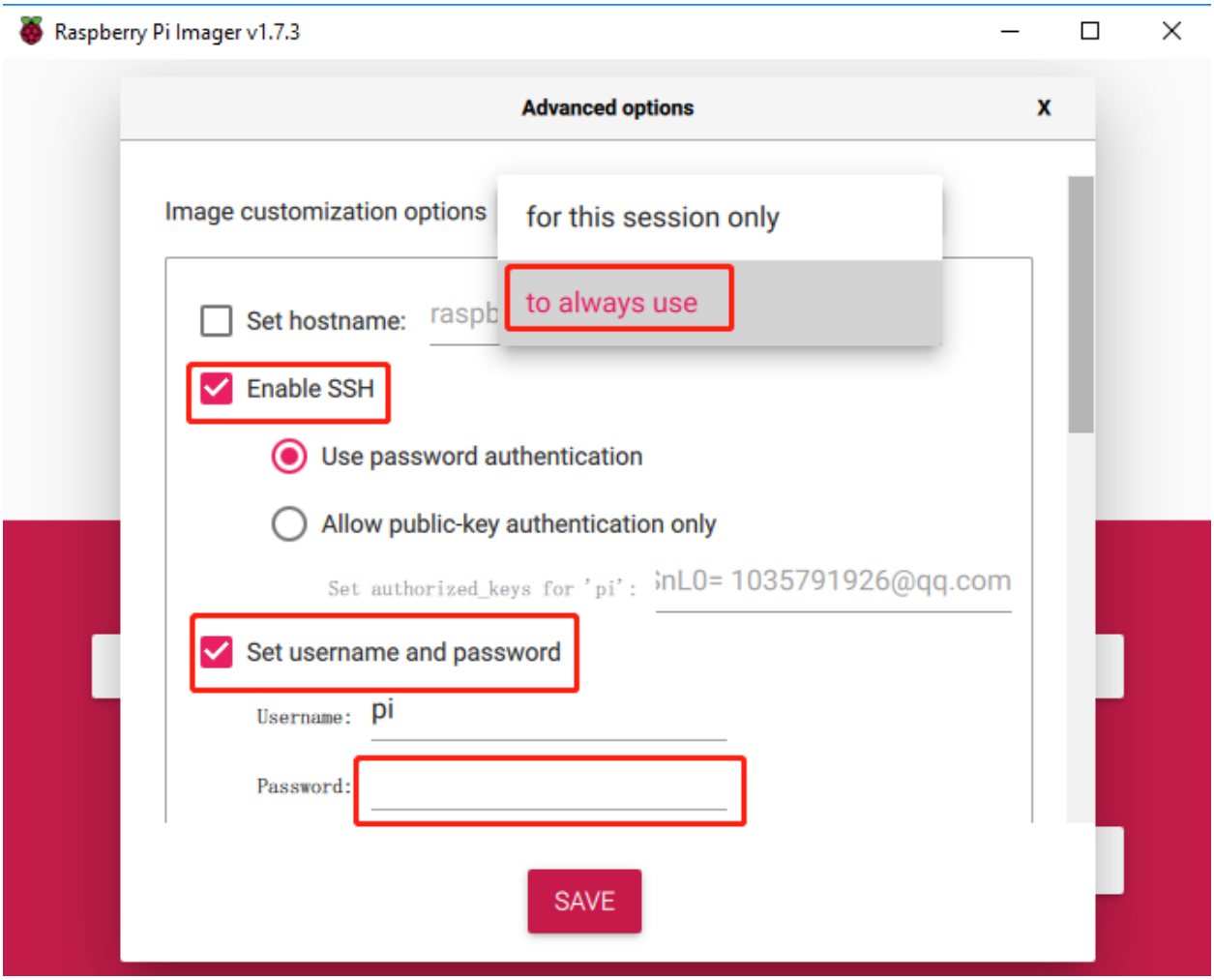


第 6 步

要打开高级选项页面，请单击设置按钮（选择操作系统后出现）或按 Ctrl+Shift+X。

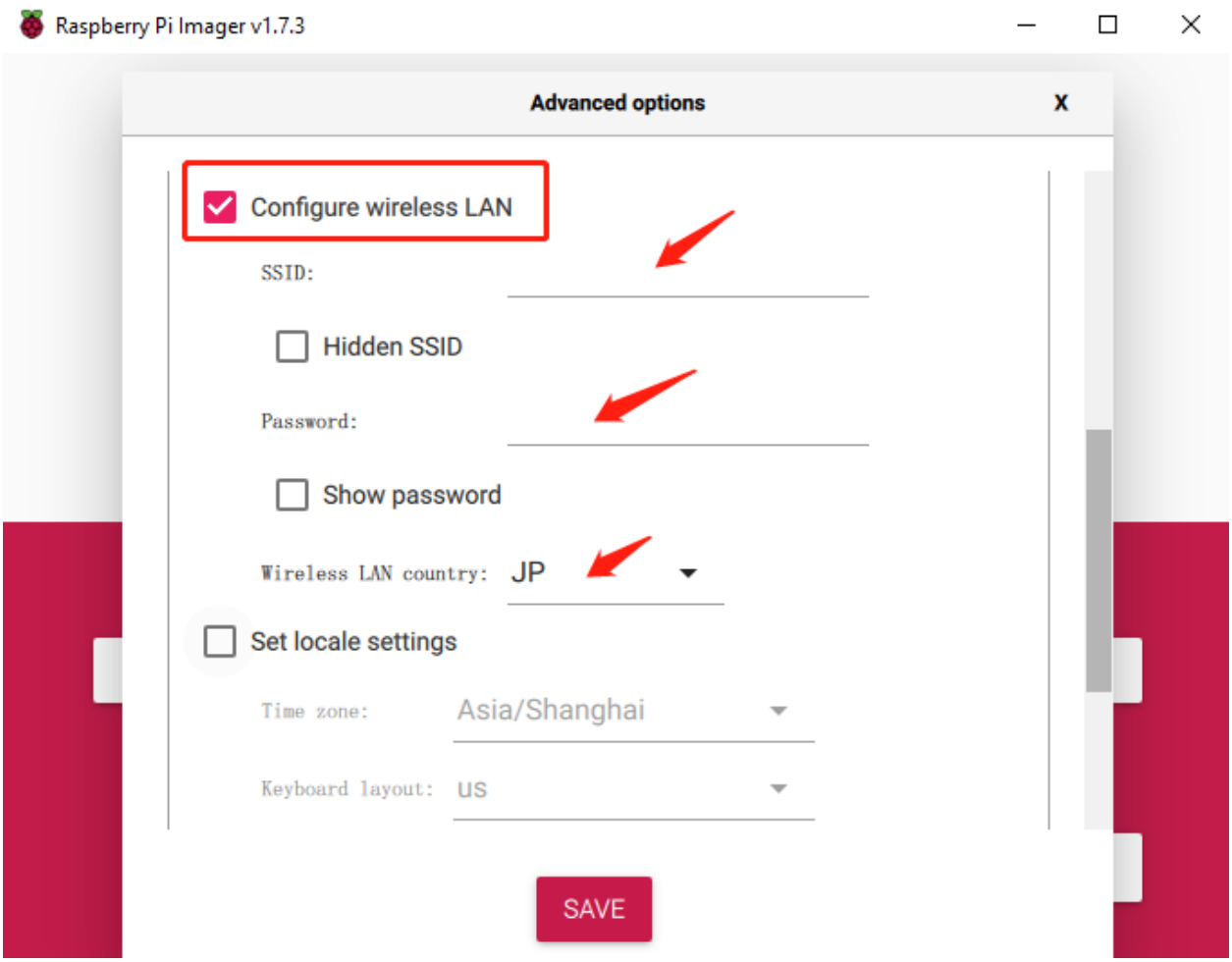
现在，设置主机名，启用 ssh 并设置用户名和密码。

> **警告：** 请务必记下 `hostname`、`username`、和 `password`; 它们对于以后远程访问 Raspberry Pi 至关重要。



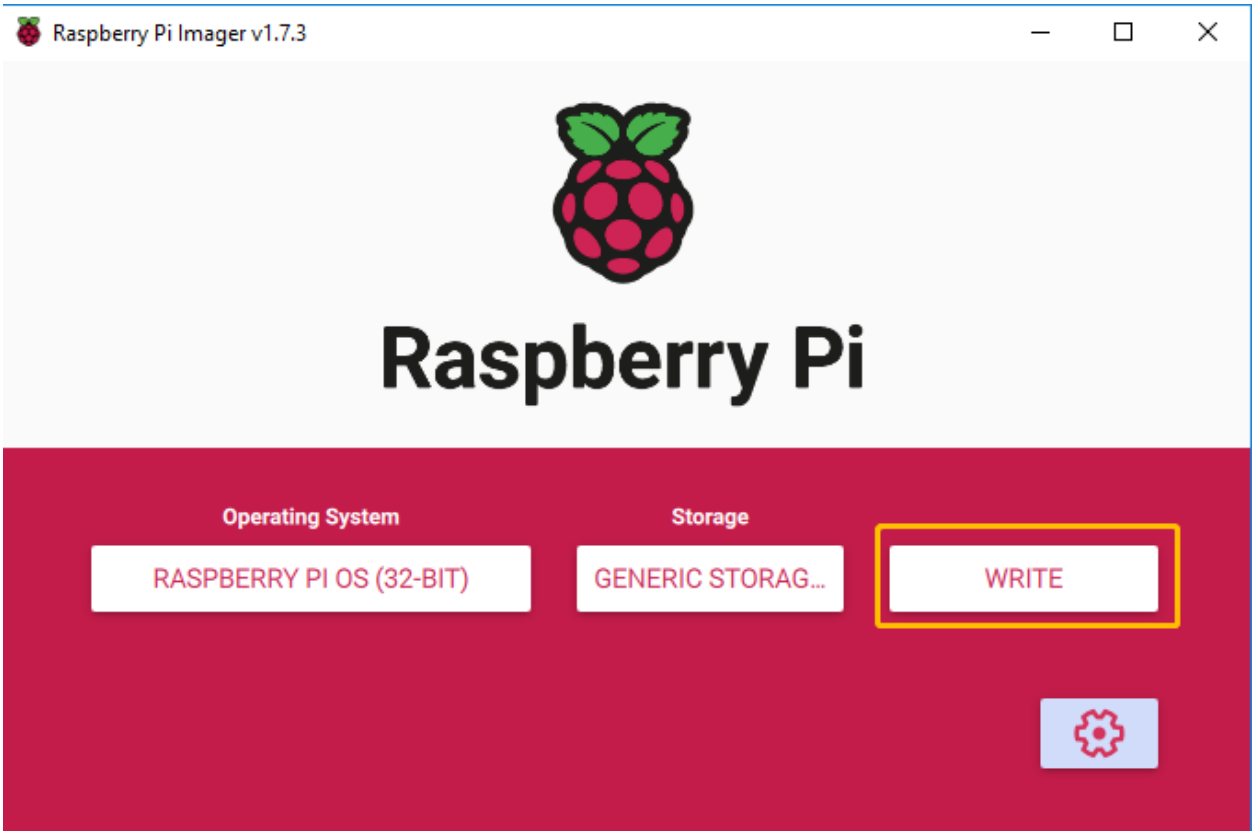
然后向下滚动以完成 wifi 配置并单击 **SAVE** 。

备注： **wifi country** 选择 **CN**。



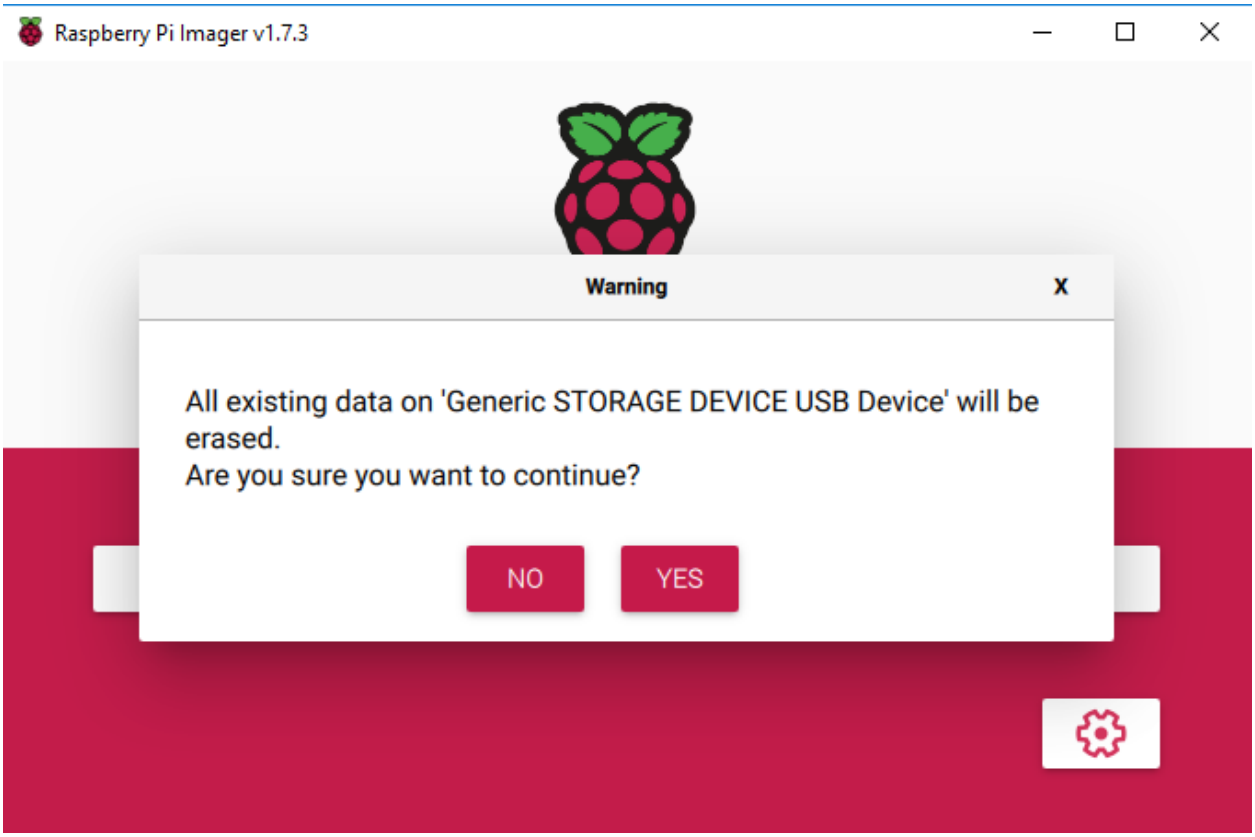
第 7 步

单击 **WRITE** 按钮。



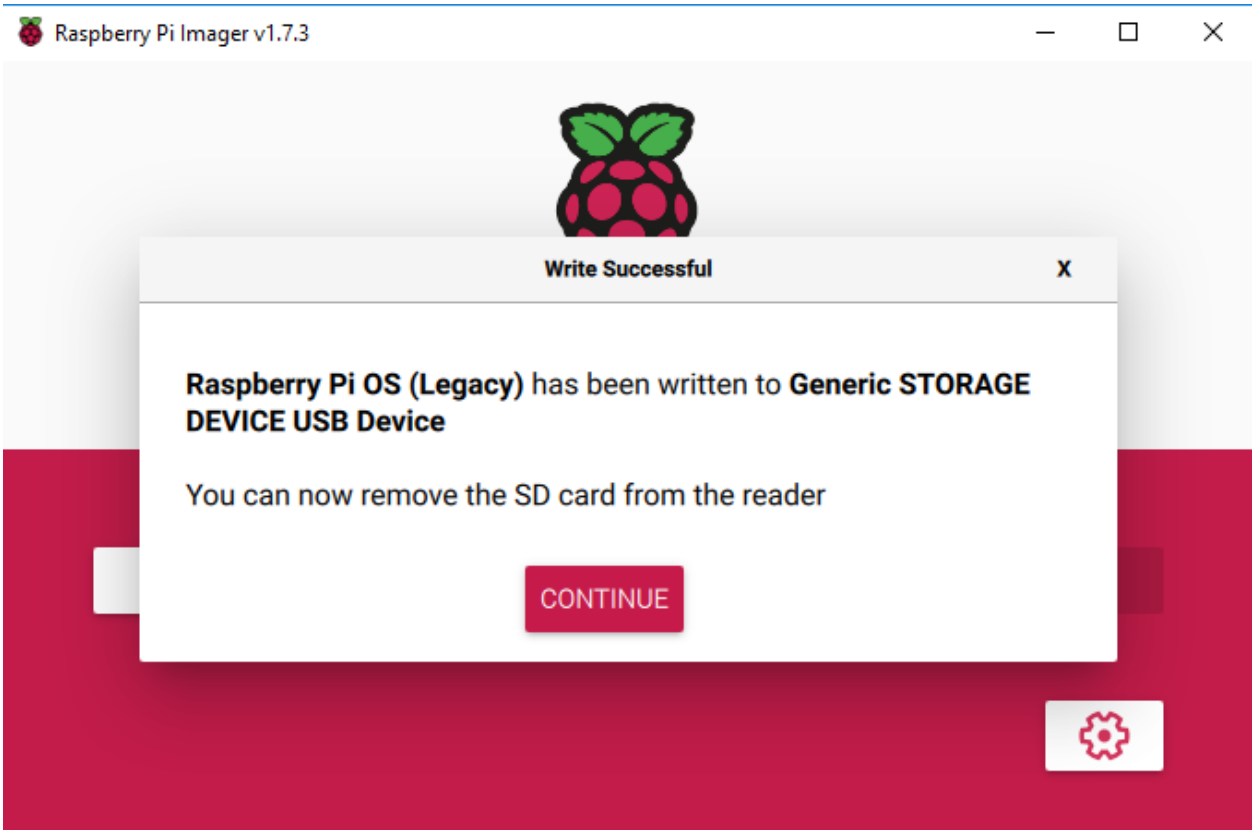
第 8 步

如果您的 SD 卡上当前有任何文件，您可能希望先备份这些文件以防止永久丢失它们。如果没有要备份的文件，请单击 **YES**。



第 9 步

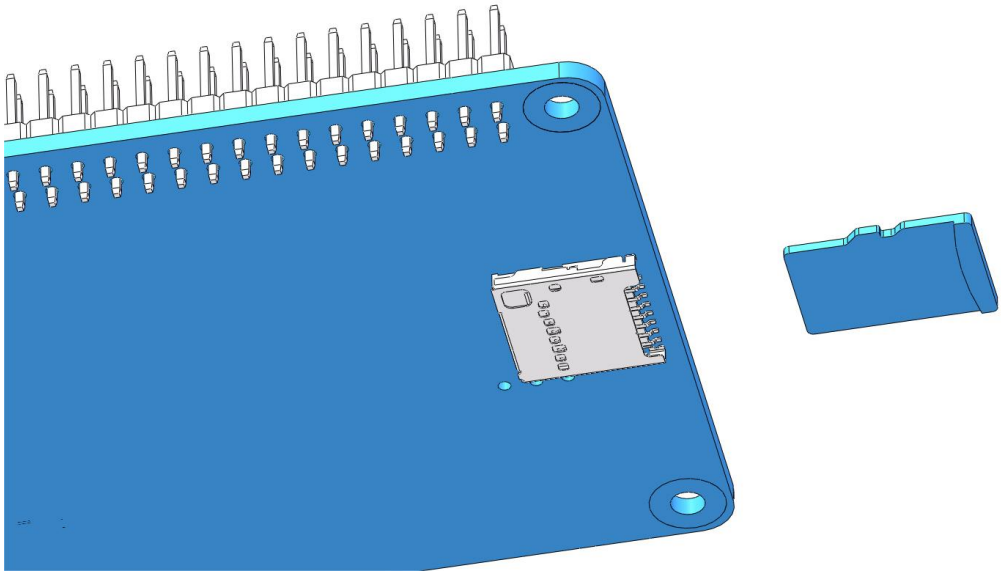
等待一段时间后，会出现如下窗口，代表写入完成。



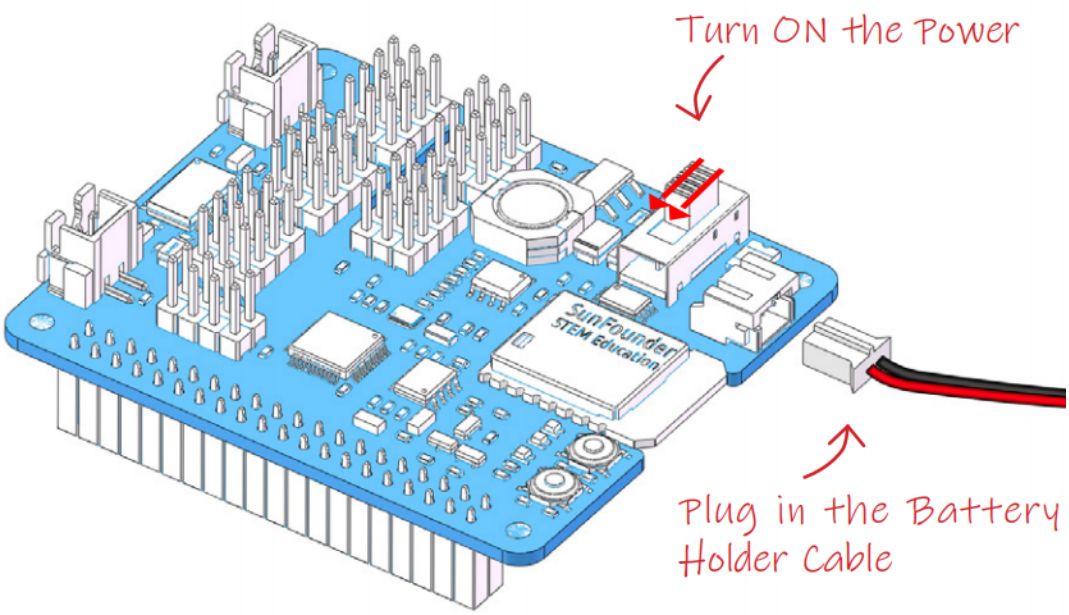
5.1.3 设置您的树莓派

为树莓派供电（重要）

1. 将已安装树莓派 OS 的 SD 卡插入树莓派底部的 microSD 卡插槽。



2. 按照组装折页上的说明，插入电池盒电缆并将电源开关拨到 ON。



如果您有屏幕

如果你有屏幕，操作树莓派对你来说会很容易。

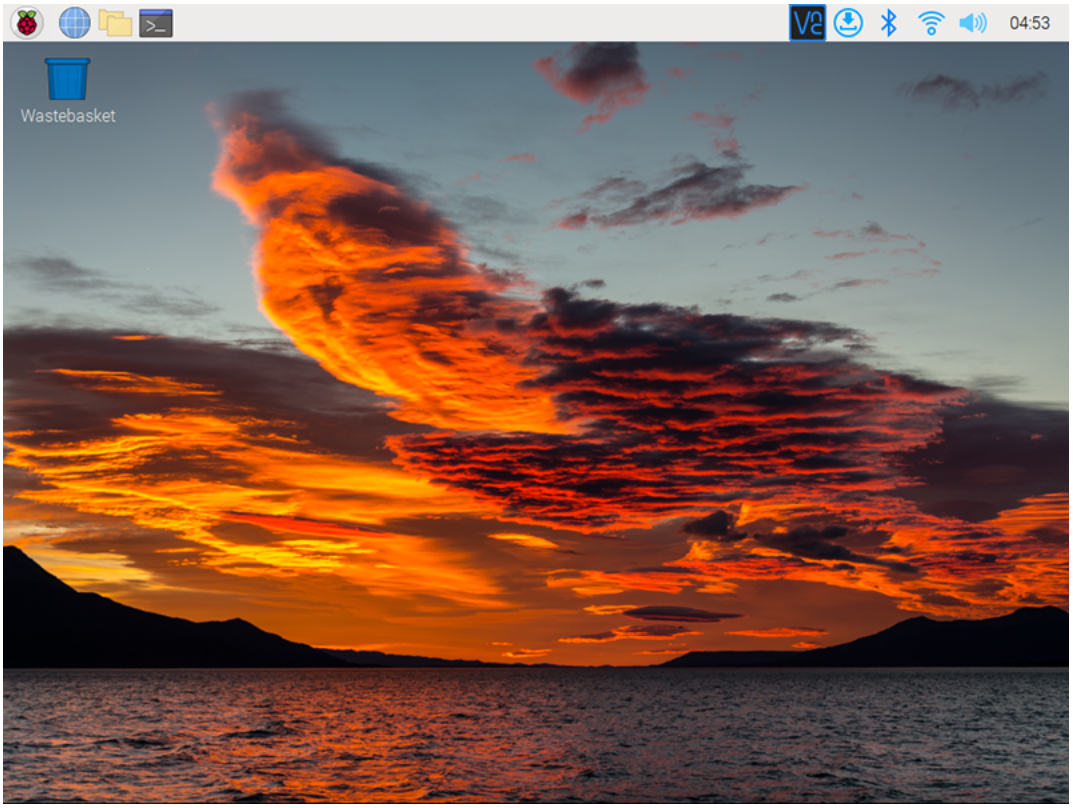
所需组件

- 任意型号的树莓派
- 1 * 电源适配器
- 1 * Micro SD 卡
- 1 * 屏幕电源适配器
- 1 * HDMI 线
- 1 * 屏幕
- 1 * 鼠标
- 1 * 键盘

1. 插入鼠标和键盘。
2. 将屏幕连接到树莓派的 HDMI 端口，并确保你的屏幕插入墙壁插座并已开机。

> **备注：** 如果你使用的是树莓派 4，你需要将屏幕连接到 HDMI0 端口（靠近电源输入端口的位置）。

3. 使用电源适配器为树莓派供电。几秒钟后，树莓派 OS 桌面将会显示。



如果您没有屏幕

如果您没有显示器，您可以远程登录到您的树莓派。

您可以使用 SSH 命令来打开树莓派的 Bash shell。Bash 是 Linux 的默认标准 shell。shell 本身是用户在使用 Unix/Linux 时的命令（指令）。您需要做的大部分操作都可以通过 shell 完成。

如果你不满意使用命令窗口来访问你的树莓派，你也可以使用远程桌面功能，用图形界面轻松管理树莓派上的文件。

请参阅下面的各系统详细教程。

Mac OS X 用户

对于 Mac 用户，通过 VNC 直接访问 Raspberry Pi 桌面比从命令行更方便。通过在 Raspberry Pi 端启用 VNC 后，可以通过 Finder 输入设置的账户密码来访问它。

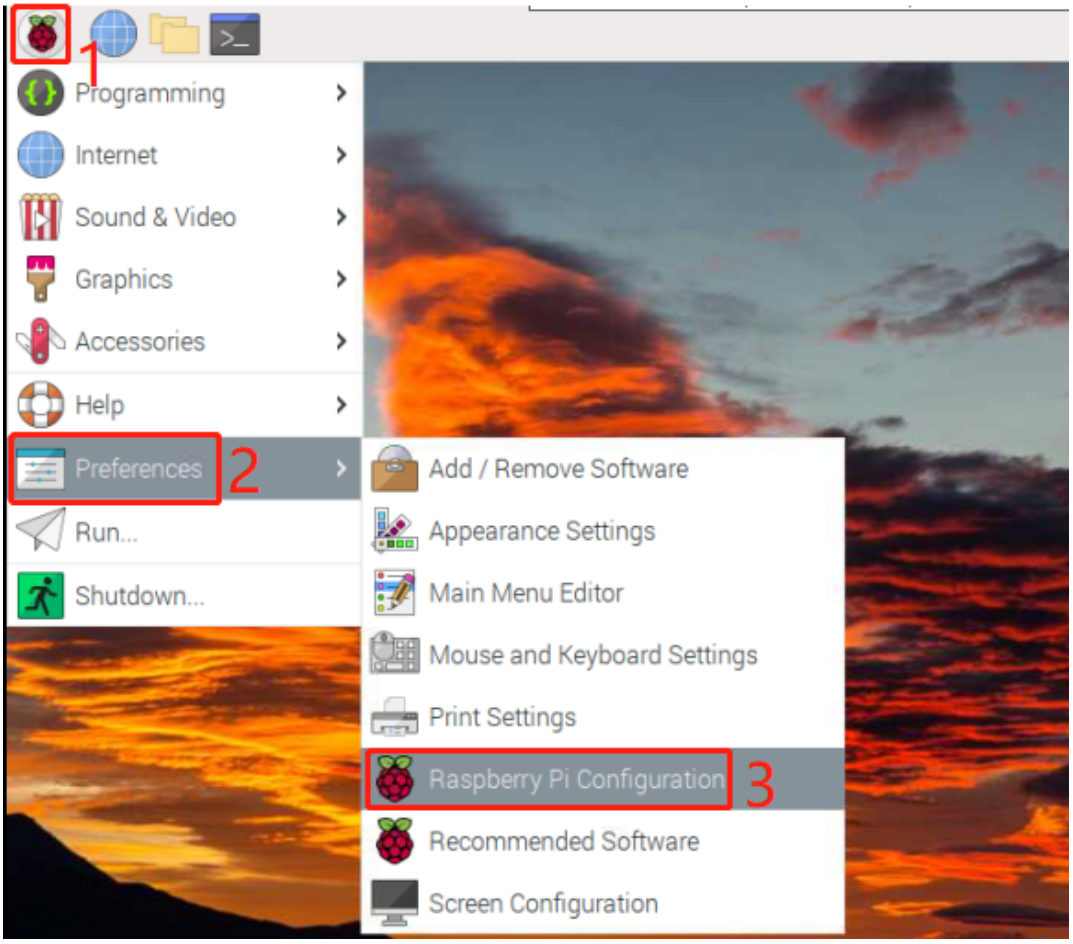
请注意，此方法不会加密 Mac 与 Raspberry Pi 之间的通信。通信将在您的家庭或商业网络内进行，所以即使它是不受保护的，也不会有问题。但是，如果你担心这个问题，你可以安装像 VNC® Viewer 这样的 VNC 应用程序。

另外，如果您可以使用临时的显示器（电视）、鼠标和键盘直接打开 Raspberry Pi 桌面以设置 VNC，那将非常方便。如果没有，也没关系，您也可以使用 SSH 命令打开 Raspberry Pi 的 Bash shell，然后使用命令设置 VNC。

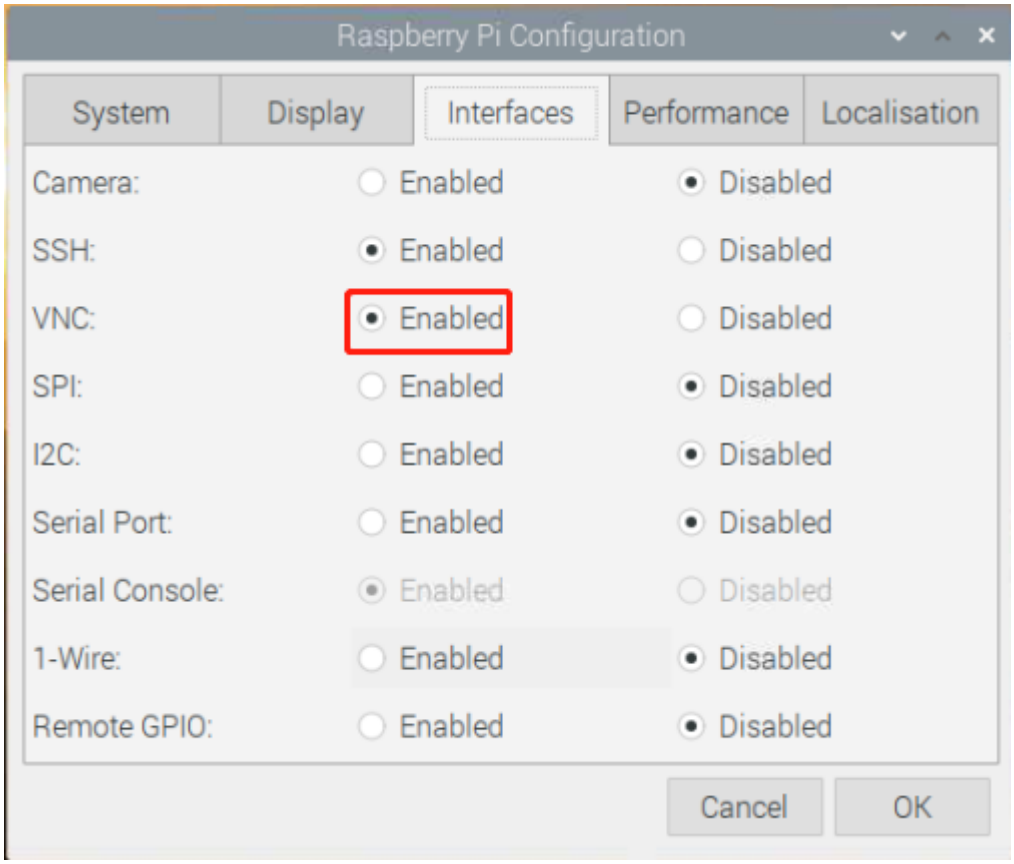
- 有临时显示器（或电视）吗？
- 没有临时显示器（或电视）吗？

有临时显示器（或电视）吗？

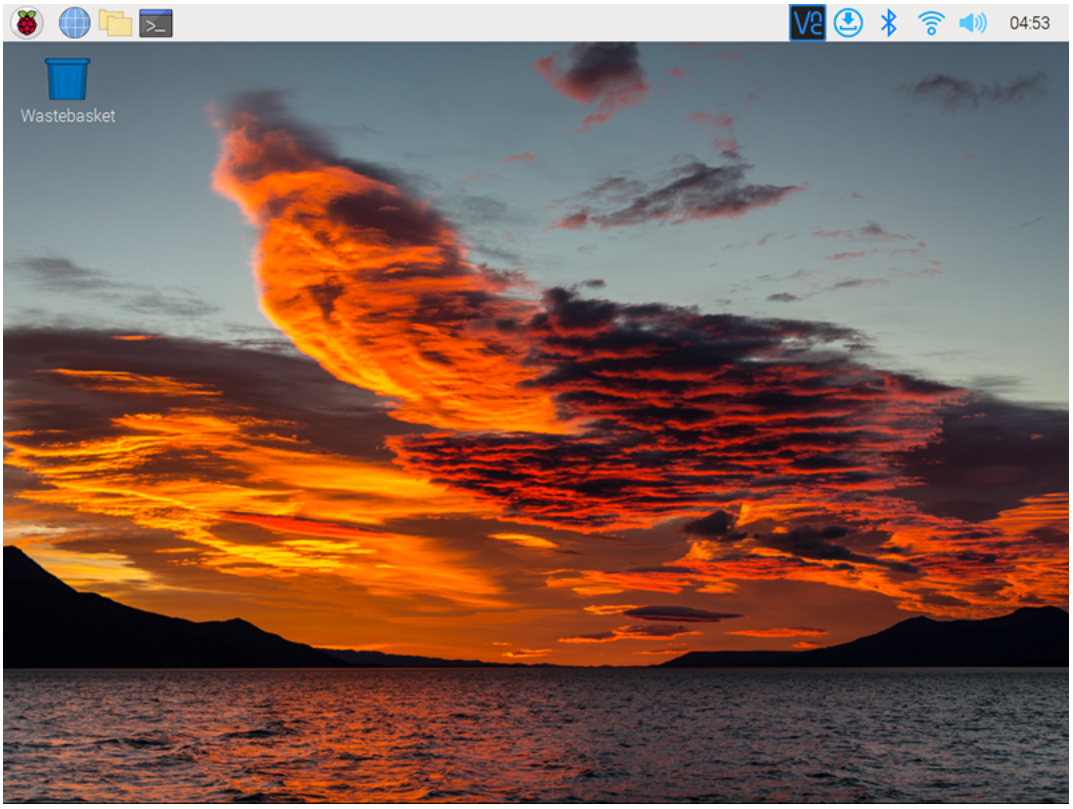
1. 将显示器（或电视）、鼠标和键盘连接到 Raspberry Pi 并开机。根据图中的数字选择菜单。



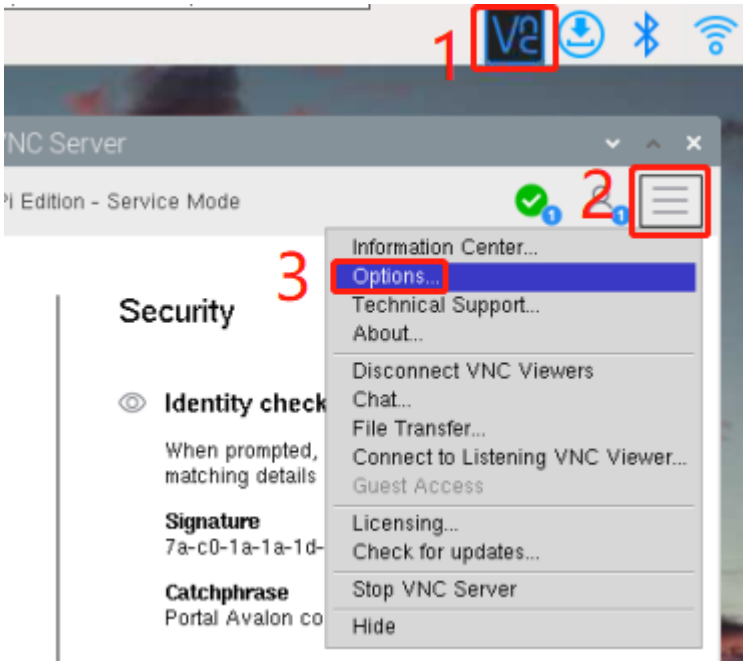
2. 将显示以下屏幕。在 **Interfaces** 选项卡上将 **VNC** 设置为 **Enabled**，然后点击 **OK**。



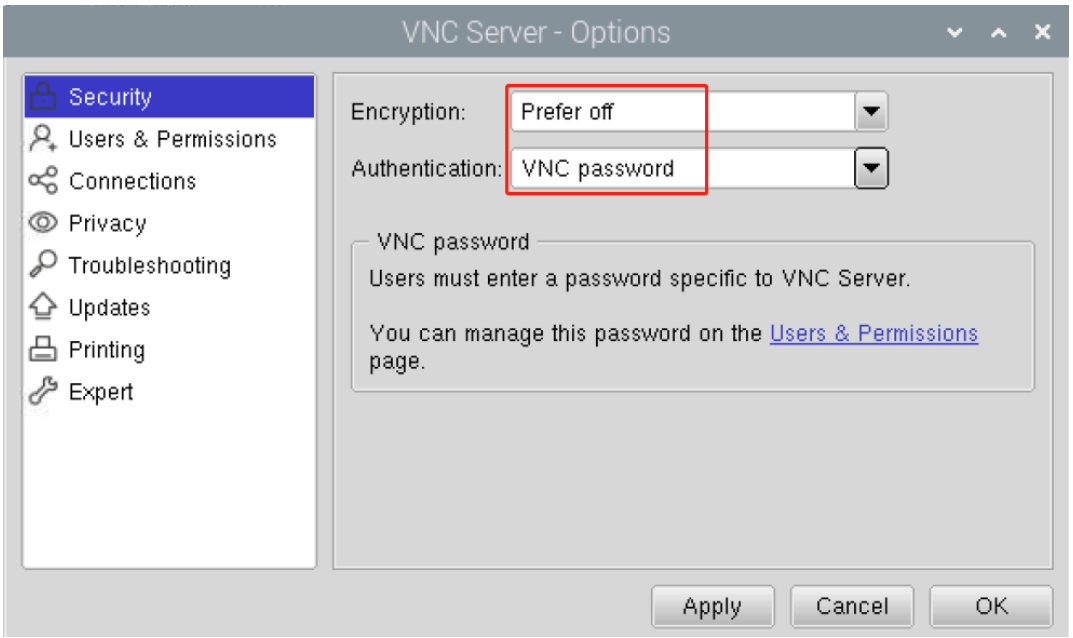
3. VNC 图标出现在屏幕的右上方，VNC 服务器启动。



4. 通过点击 **VNC** 图标打开 VNC 服务器窗口，然后点击右上角的 **Menu** 按钮并选择 **Options**。

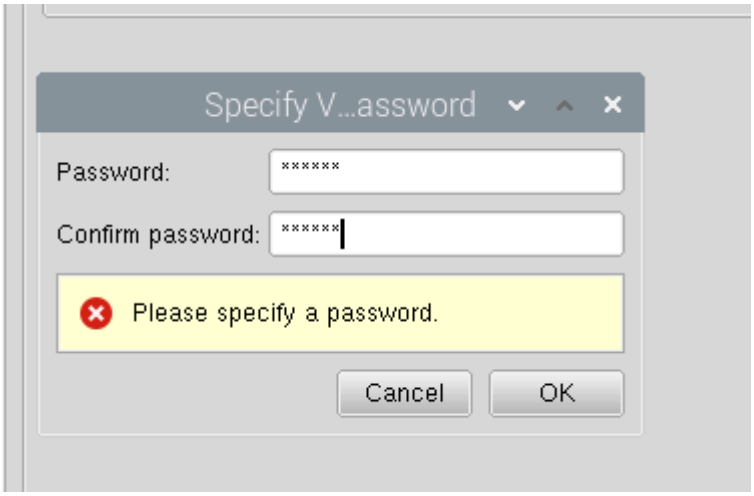


5. 您将看到以下屏幕，您可以在此更改选项。



将 **Encryption** 设置为 **Prefer off**，将 **Authentication** 设置为 **VNC password**。

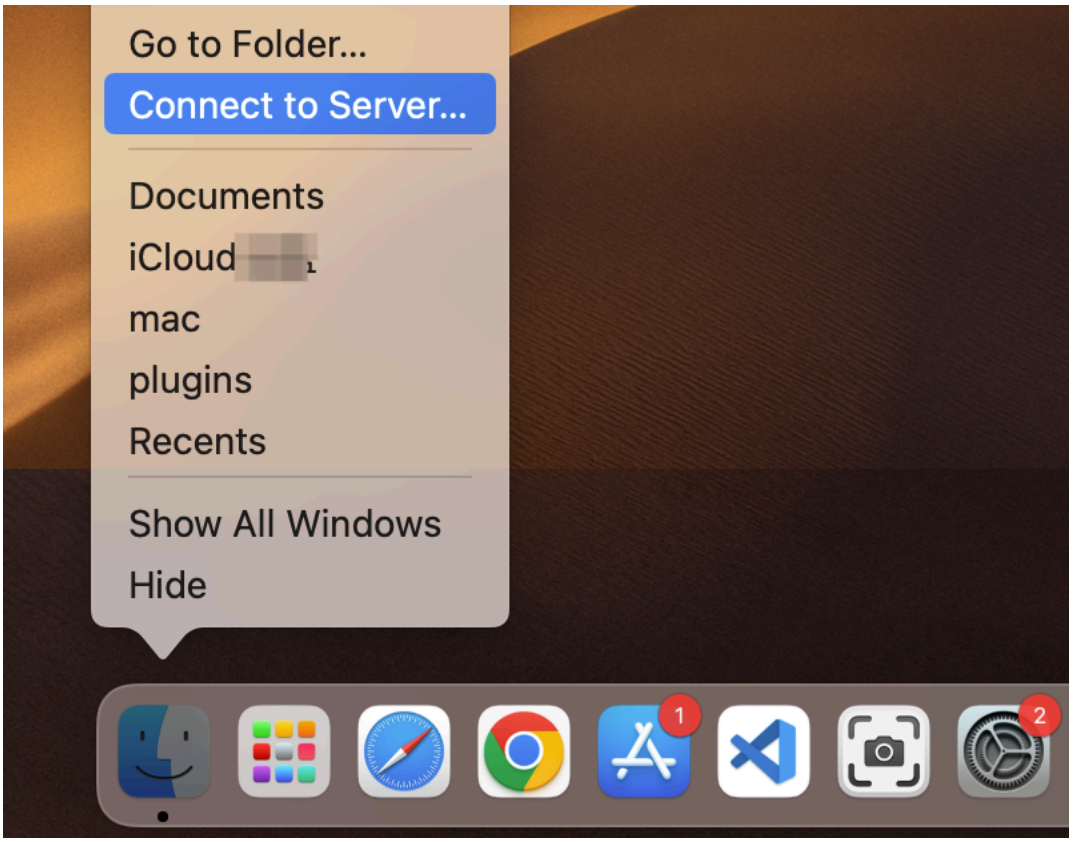
6. 当您点击 **OK** 按钮时，会显示密码输入屏幕。您可以使用与 Raspberry pi 密码相同的密码或不同的密码，输入后点击 **OK**。



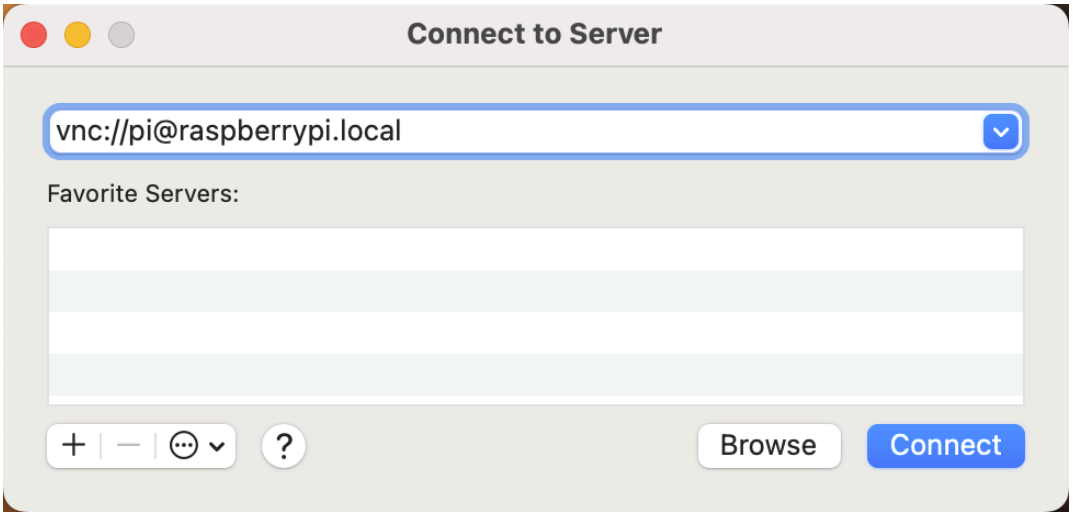
您现在可以从 Mac 连接了。可以断开显示器了。

从这里开始，将在 Mac 端操作。

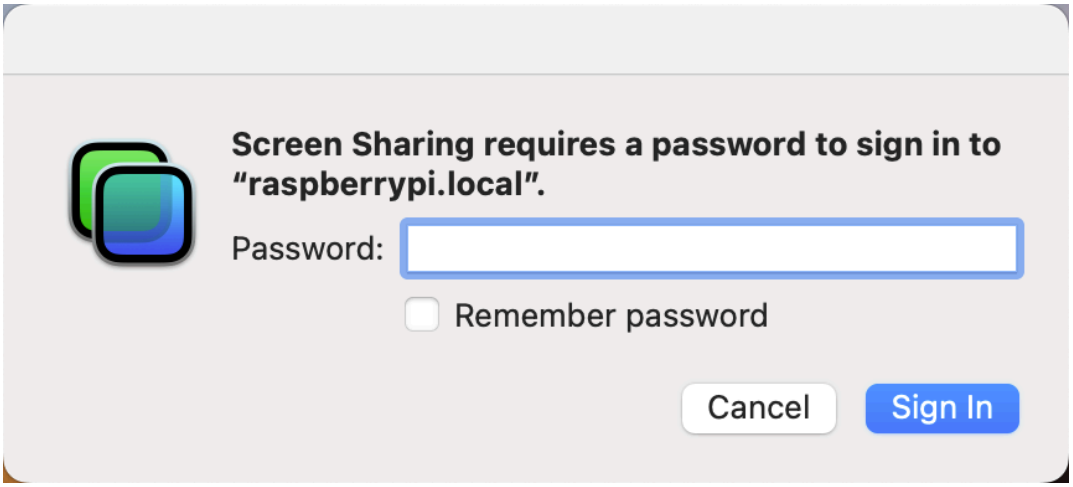
1. 现在，在 Finder 的菜单中选择 **Connect to Server**，可以通过右键单击打开。



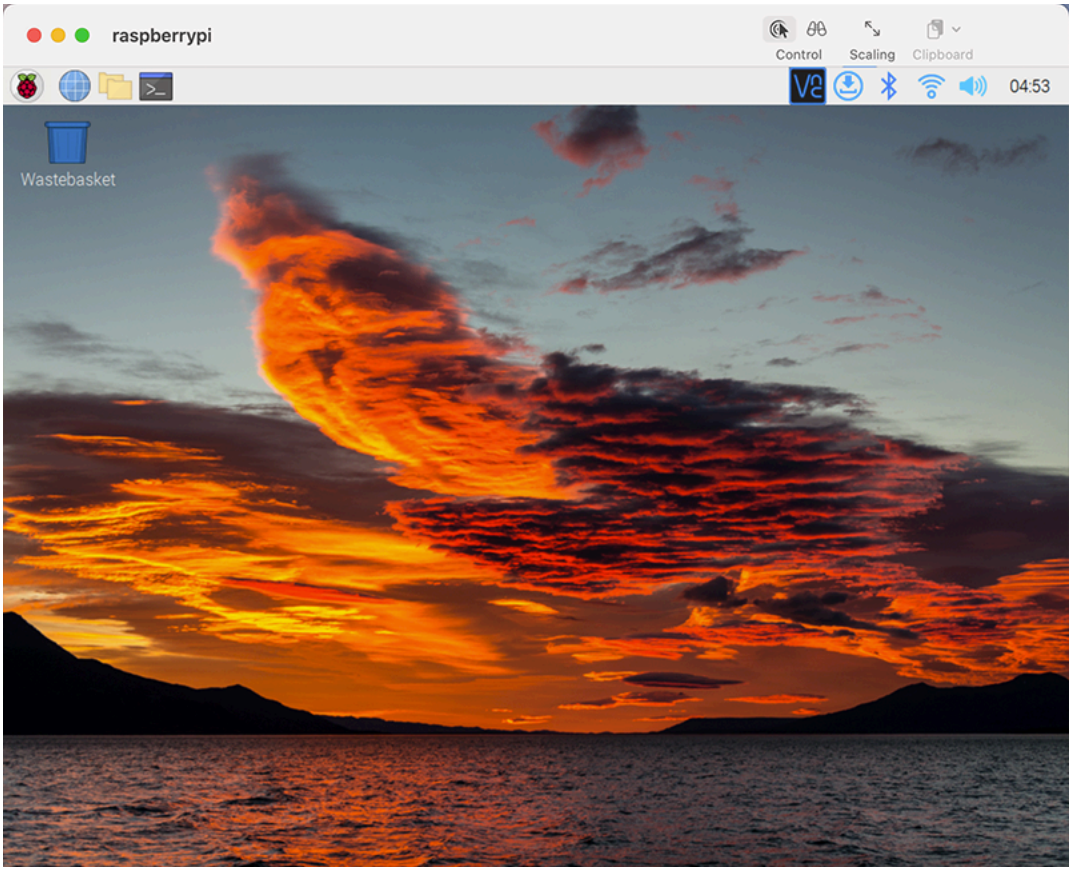
2. 输入 `vnc://< 用户名>@< 主机名>.local` （或 `vnc://< 用户名>@<IP 地址>`）。输入后，点击 **Connect**。



3. 会要求您输入密码，请输入。



4. Raspberry pi 的桌面将被显示，您将能够从 Mac 上操作它。

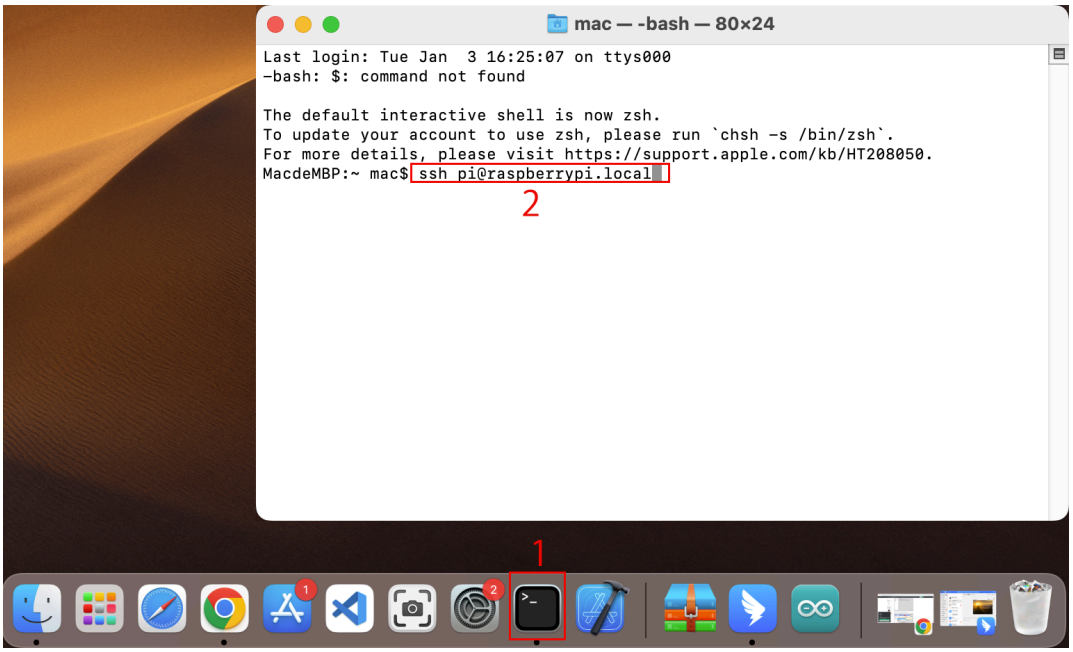


没有临时显示器（或电视）吗？

- 您可以应用 SSH 命令打开 Raspberry Pi 的 Bash shell。
- Bash 是 Linux 的标准默认 shell。
- shell 本身是用户使用 Unix/Linux 时的命令（指令）。
- 您需要做的大部分操作都可以通过 shell 完成。
- 设置了 Raspberry pi 端后，您可以从 Mac 的 **Finder** 访问 Raspberry Pi 的桌面。

1. 输入 `ssh < 用户名>@< 主机名>.local` 连接到 Raspberry Pi。

```
ssh pi@raspberrypi.local
```



2. 下面的信息只会在你第一次登录时显示，所以输入 **yes**。

```
The authenticity of host 'raspberrypi.local␣
↪(2400:2410:2101:5800:635b:f0b6:2662:8cba)' can't be established.
ED25519 key fingerprint is SHA256:oo7x3ZSgAo032wD1tE8eW0fFM/
↪kmewIvRwkBys6XRwg.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

3. 输入 Raspberry pi 的密码。您输入的密码不会显示，所以请注意不要出错。

```
pi@raspberrypi.local's password:
Linux raspberrypi 5.15.61-v8+ #1579 SMP PREEMPT Fri Aug 26 11:16:44 BST␣
↪2022 aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Sep 22 12:18:22 2022
pi@raspberrypi:~ $
```

4. 一旦成功登录，设置 Raspberry Pi 以便您可以从 Mac 通过 VNC 登录。首先更新您的操作系统，运行以下命令。

```
sudo apt update
sudo apt upgrade
```

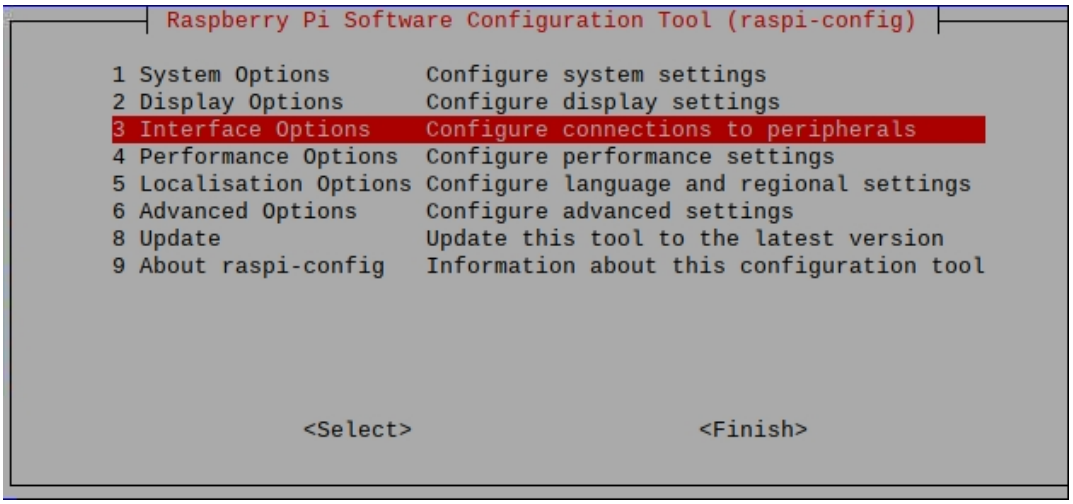
当提示 `Do you want to continue? [Y/n]` 时，输入 `Y`。

更新可能需要一段时间才能完成。(这取决于那时的更新量。)

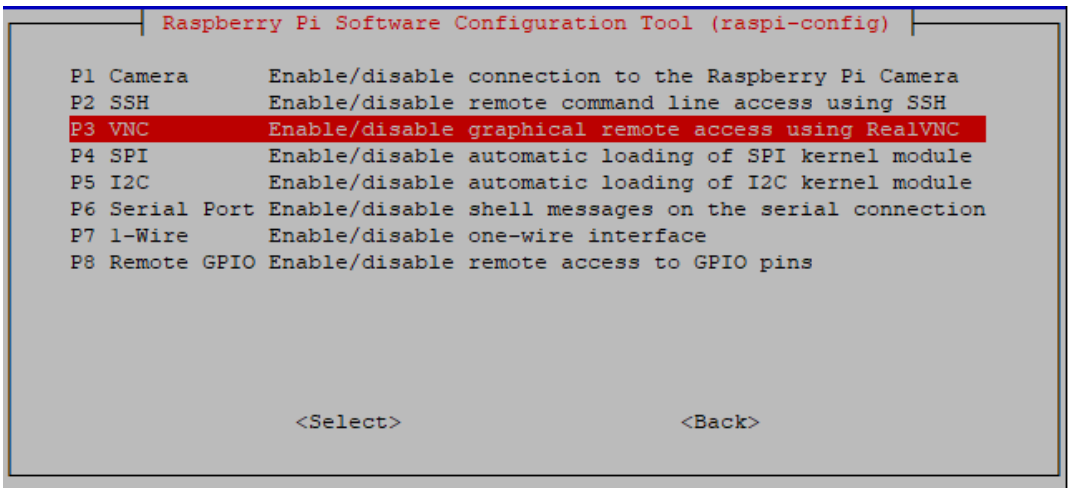
5. 输入以下命令以启用 **VNC 服务器**。

```
sudo raspi-config
```

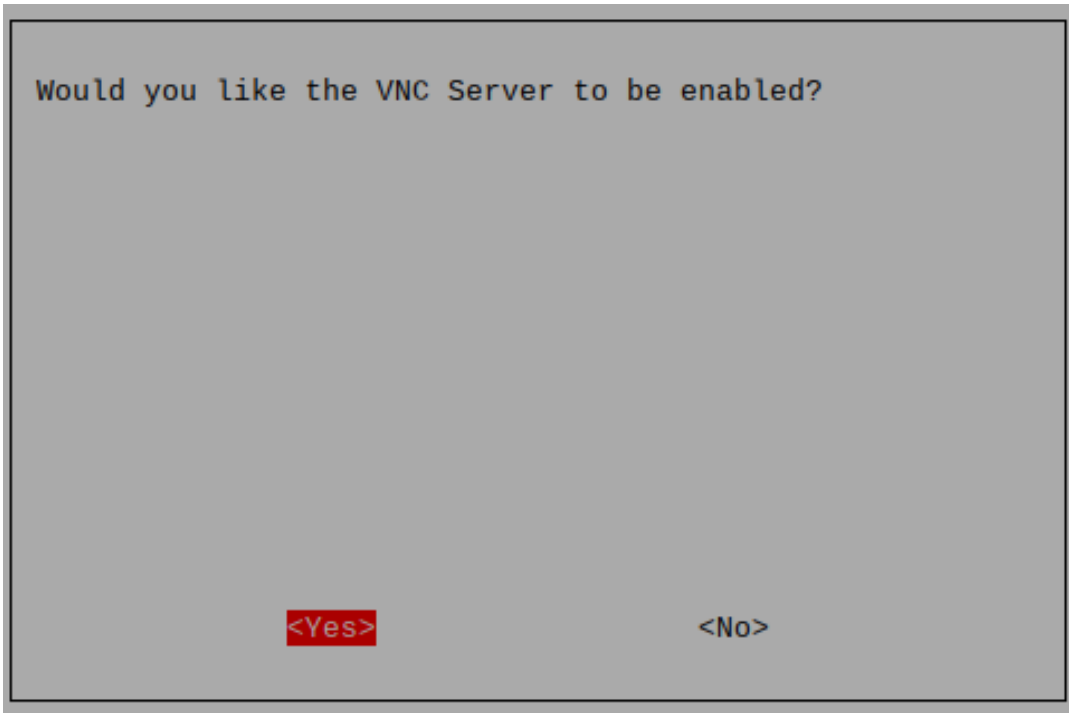
6. 将显示以下屏幕。使用键盘上的箭头键选择 **3 Interface Options**，然后按 **Enter** 键。



7. 然后选择 **VNC**。



8. 使用键盘上的箭头键选择 **<Yes>** -> **<OK>** -> **<Finish>** 完成设置。

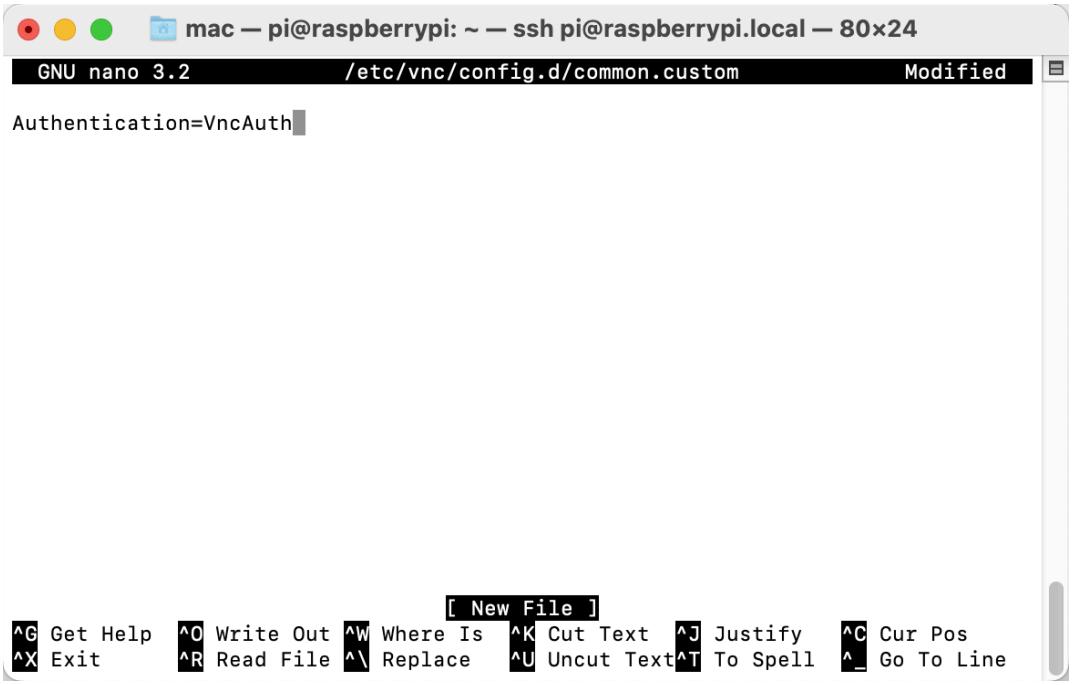


9. 现在 VNC 服务器已经启动，让我们更改从 Mac 连接的设置。

   要为计算机上的所有用户账户的所有程序指定参数，请创建 `/etc/vnc/config.d/common.custom`。

```
sudo nano /etc/vnc/config.d/common.custom
```

   输入 `Authentication=VncAuthenter` 后，按 `Ctrl+X` -> `Y` -> `Enter` 保存并退出。



10. 此外，设置一个密码以便从 Mac 通过 VNC 登录。您可以使用与 Raspberry pi 密码相同的密码或不同的

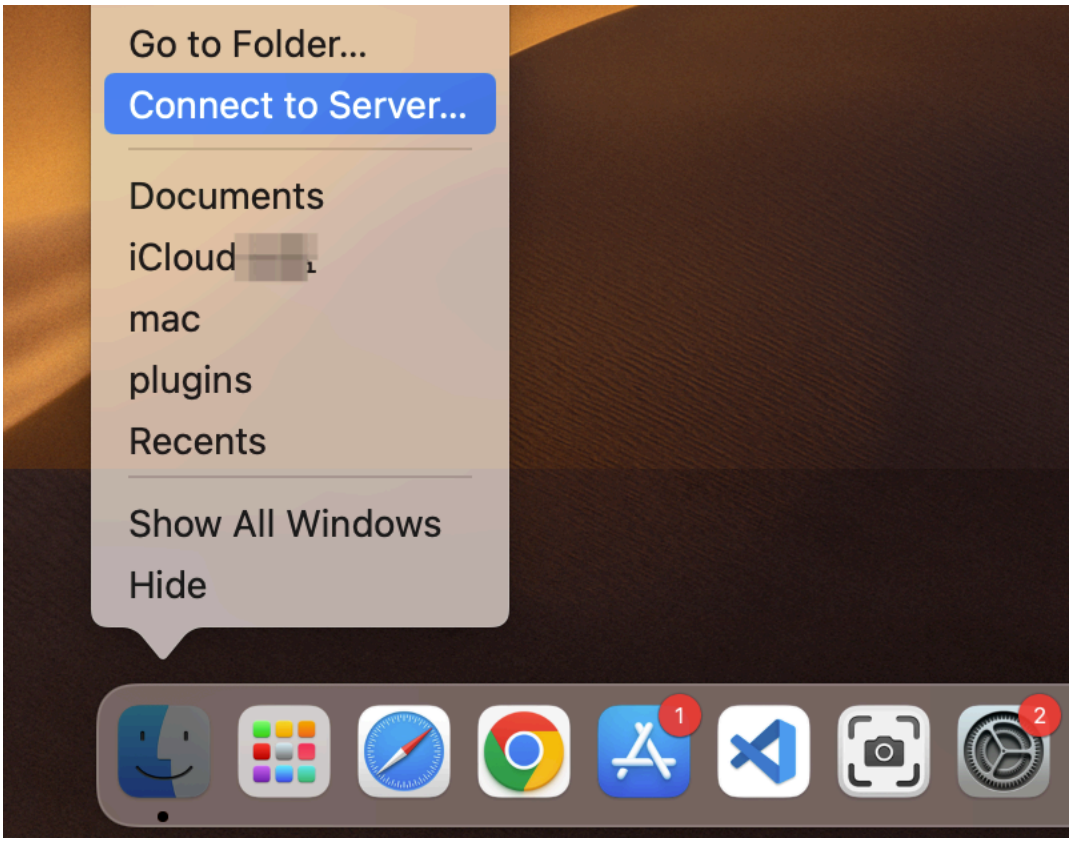
密码。

```
sudo vncpasswd -service
```

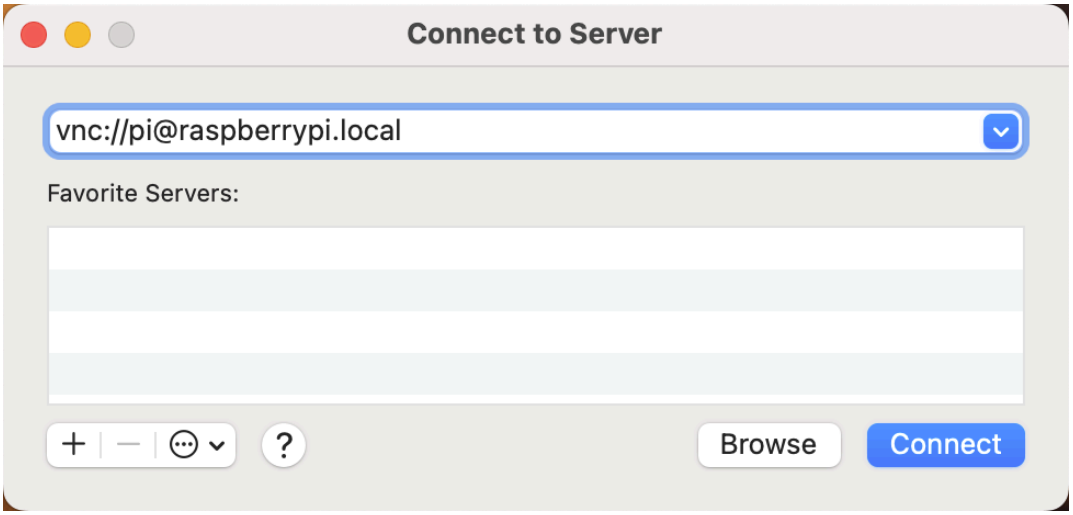
11. 一旦设置完成，重新启动 Raspberry Pi 应用更改。

```
sudo sudo reboot
```

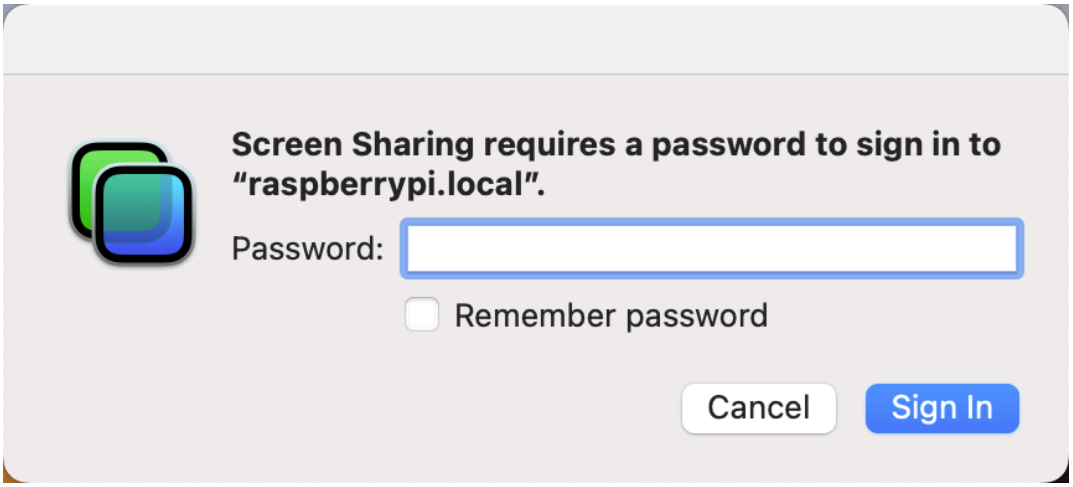
12. 现在，在 Finder 的菜单中选择 **Connect to Server**，可以通过右键单击打开。



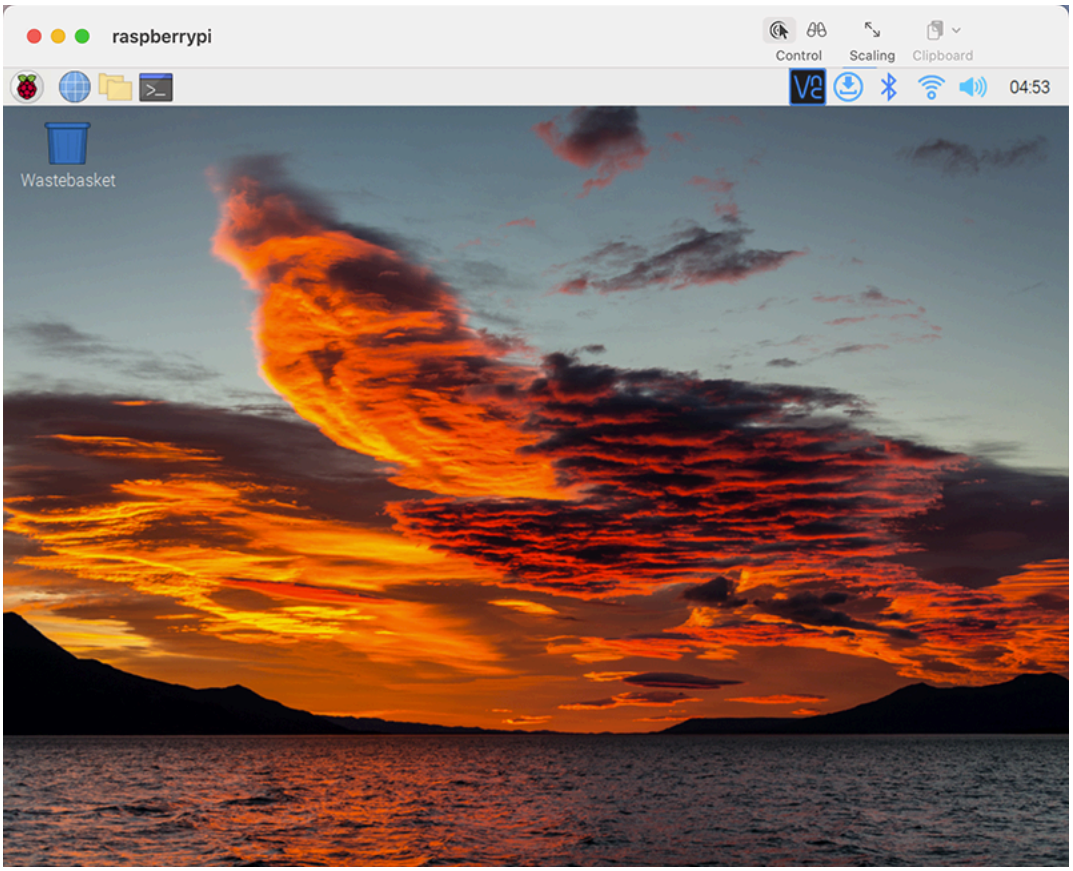
13. 输入 `vnc://< 用户名>@< 主机名>.local` （或 `vnc://< 用户名>@<IP 地址>`）。输入后，点击 **Connect**。



14. 会要求您输入密码，请输入。



15. Raspberry pi 的桌面将被显示，您将能够从 Mac 上操作它。

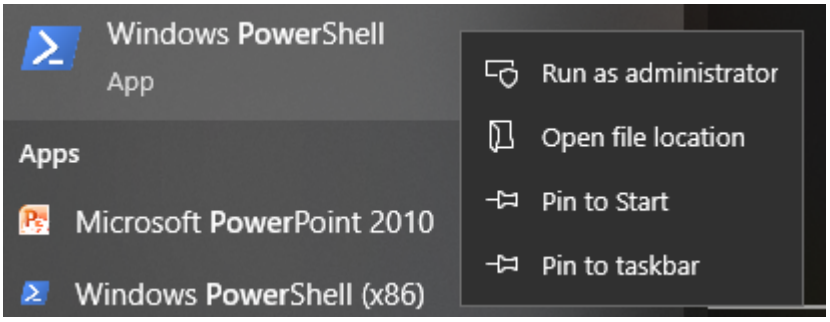


Windows 用户

远程登录树莓派

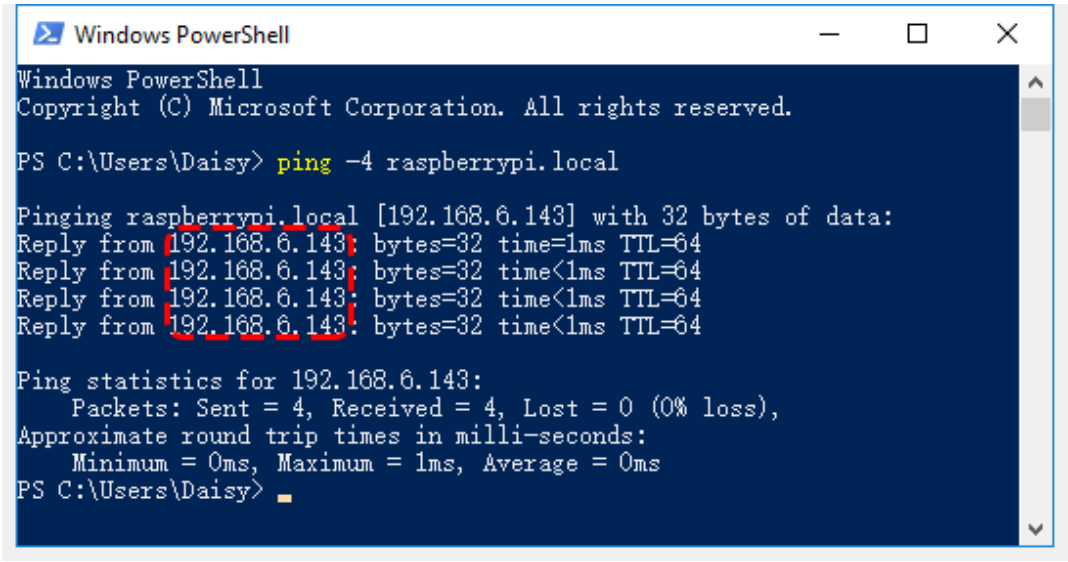
如果你正在使用 Windows 10，你可以按照以下方式远程登录树莓派。

1. 在 Windows 桌面的搜索框中输入 `powershell`，右键点击 `Windows PowerShell`，然后从弹出的菜单中选择 `以管理员身份运行`。



2. 接着，输入 `ping -4 < 主机名>.local` 来查询你的树莓派的 IP 地址。

```
ping -4 raspberrypi.local
```



如上图所示，一旦树莓派连接到网络，你就可以看到它的 IP 地址。

- 如果终端提示 `Ping request could not find host pi.local. Please check the name and try again.`，请按照提示确保您输入的主机名是正确的。
- 如果还是无法获得 IP 地址，请检查树莓派上的网络或 WiFi 配置。

3. 此时，你可以使用 `ssh < 用户名>@< 主机名>.local`` (或 ``ssh < 用户名>@<IP 地址>`) 登录到你的树莓派。

```
ssh pi@raspberrypi.local
```

> **警告：** 如果出现提示 `The term 'ssh' is not recognized as the name of a cmdlet...`。
>
> 这意味着你的系统版本较旧，没有预装 ssh 工具，你需要手动安装通过 *Powershell 安装 OpenSSH*。
>
> 或者使用第三方工具，例如*PuTTY*。

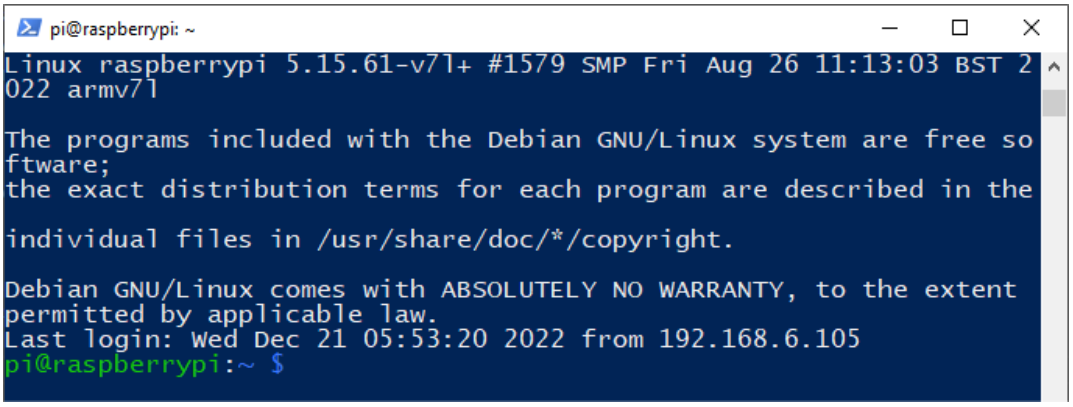
4. 当你首次登录时，会显示以下消息，请输入 `yes`。

```
The authenticity of host 'raspberrypi.local␣
↪(2400:2410:2101:5800:635b:f0b6:2662:8cba)' can't be established.
ED25519 key fingerprint is SHA256:oo7x3ZSgAo032wD1tE8eW0fFM/
↪kmewIvRwkBys6XRwg.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

5. 输入你之前设置的密码。(我的是 `raspberry`。)

> **备注：** 当你输入密码时，字符不会在窗口上显示，这是正常的。你只需输入正确的密码即可。

6. 现在，我们已经连接到了树莓派，并准备进行下一步操作。



远程桌面

如果你对使用命令窗口来访问树莓派不满意，你也可以使用远程桌面功能，用图形界面轻松管理树莓派上的文件。

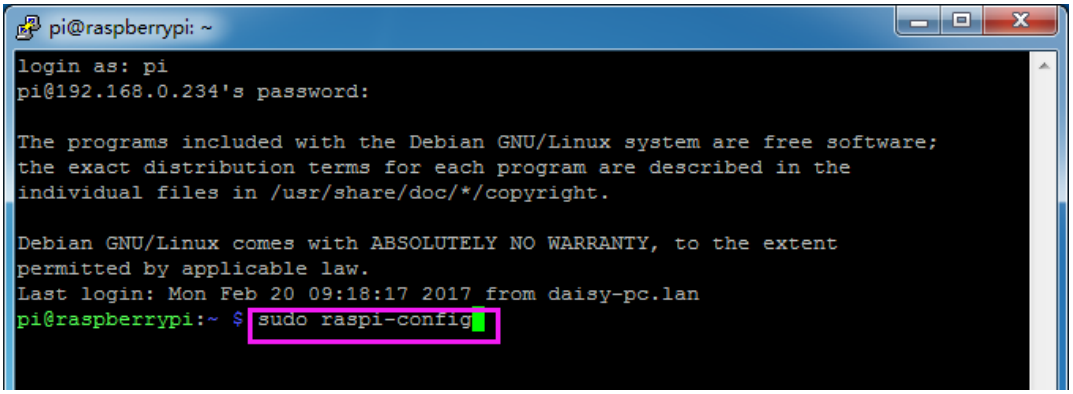
这里我们使用 VNC® Viewer。

启用 VNC 服务

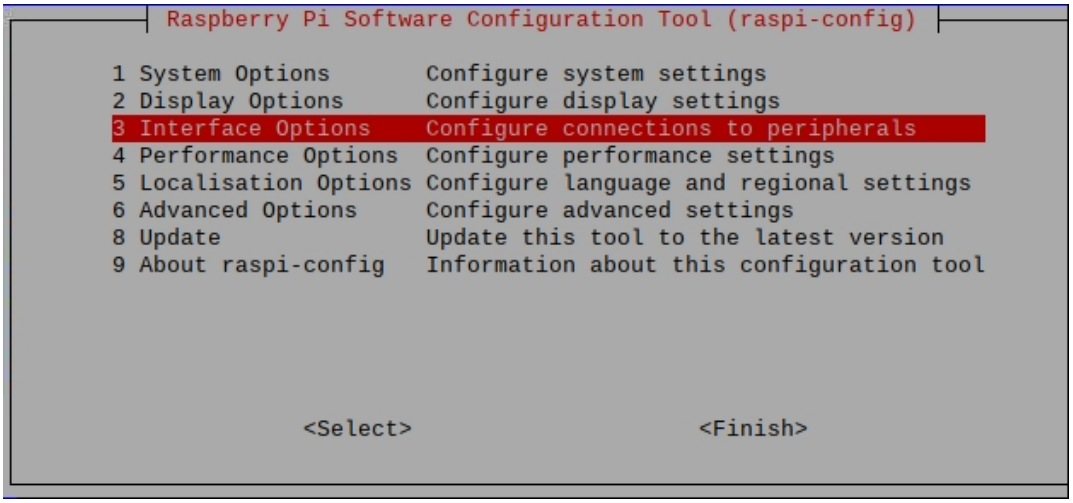
VNC 服务已经安装在系统中。默认情况下，VNC 是禁用的。你需要在配置中启用它。

1. 输入以下命令：

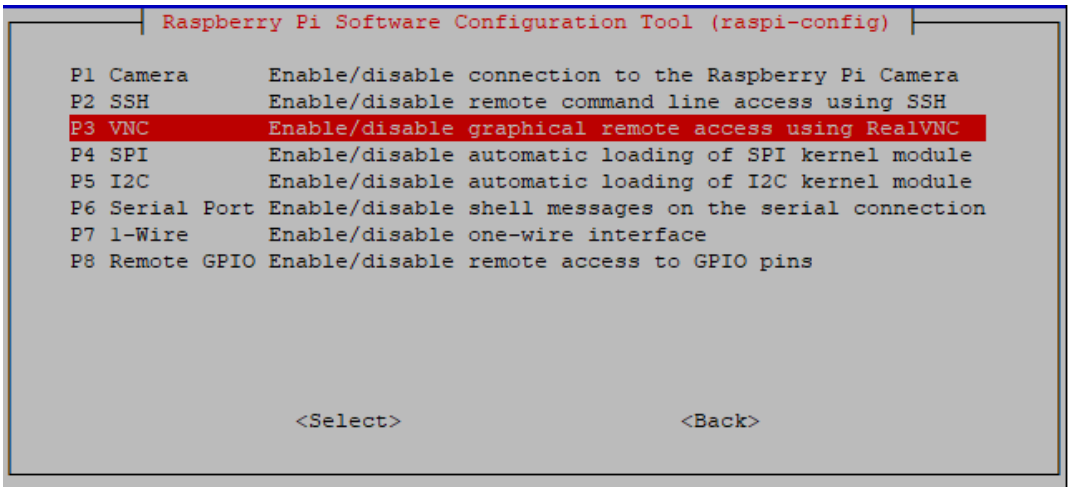
```
sudo raspi-config
```



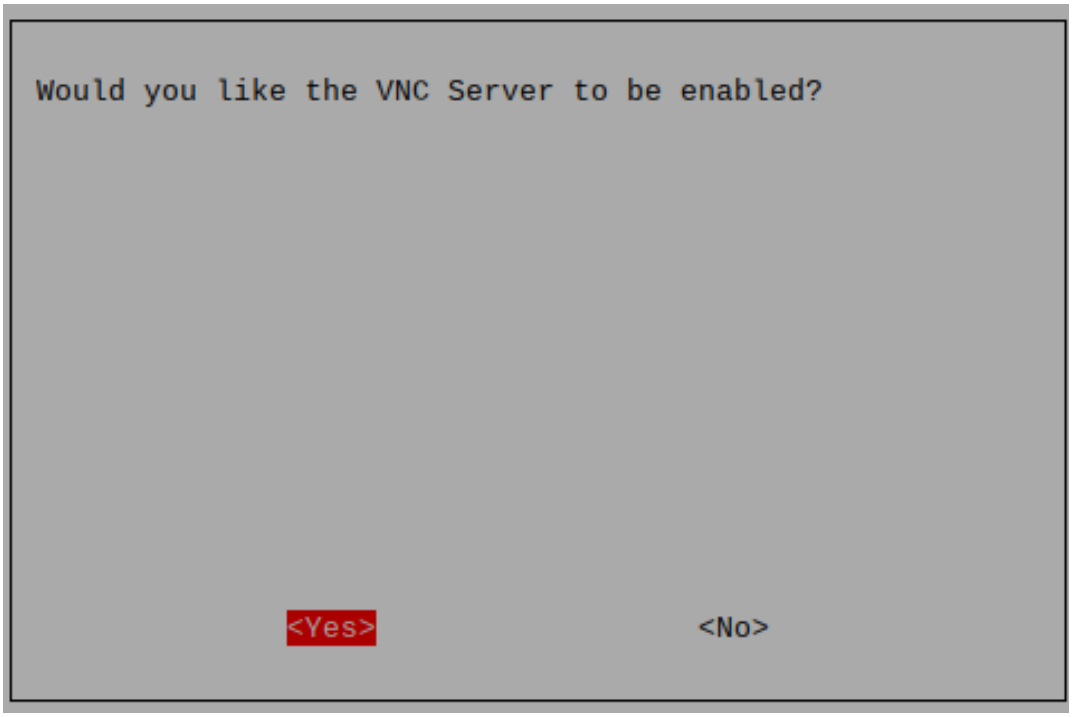
2. 使用键盘上的向下箭头键选择 **Interfacing Options**，然后按 **Enter** 键。



3. 然后选择 **VNC**。

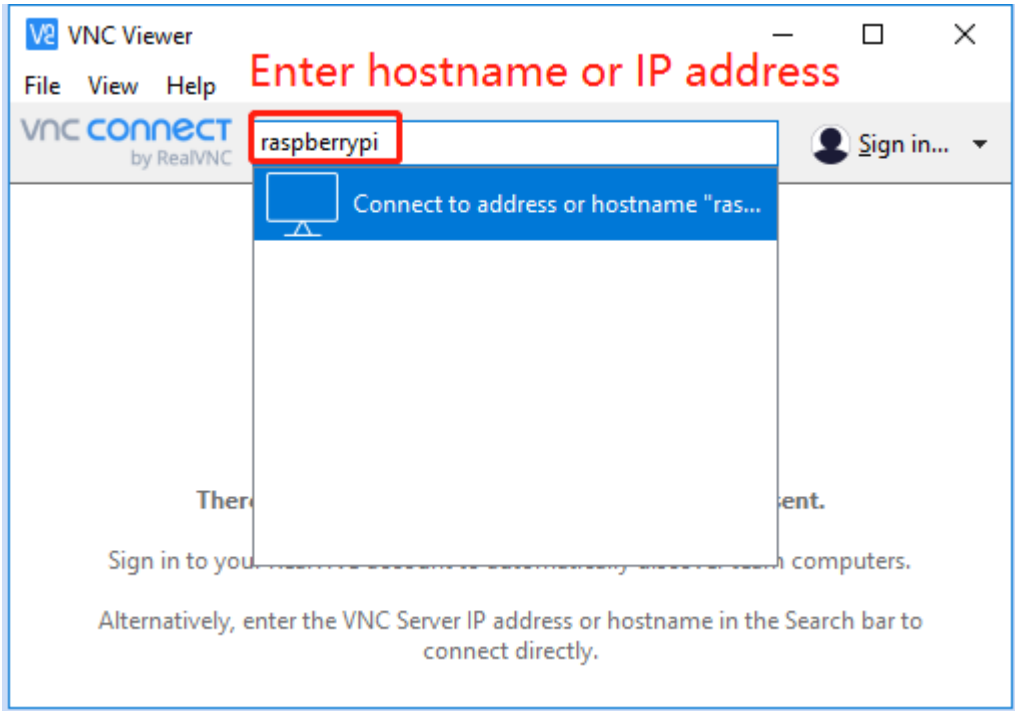


4. 使用键盘上的箭头键选择 **<Yes>** -> **<OK>** -> **<Finish>** 完成设置。

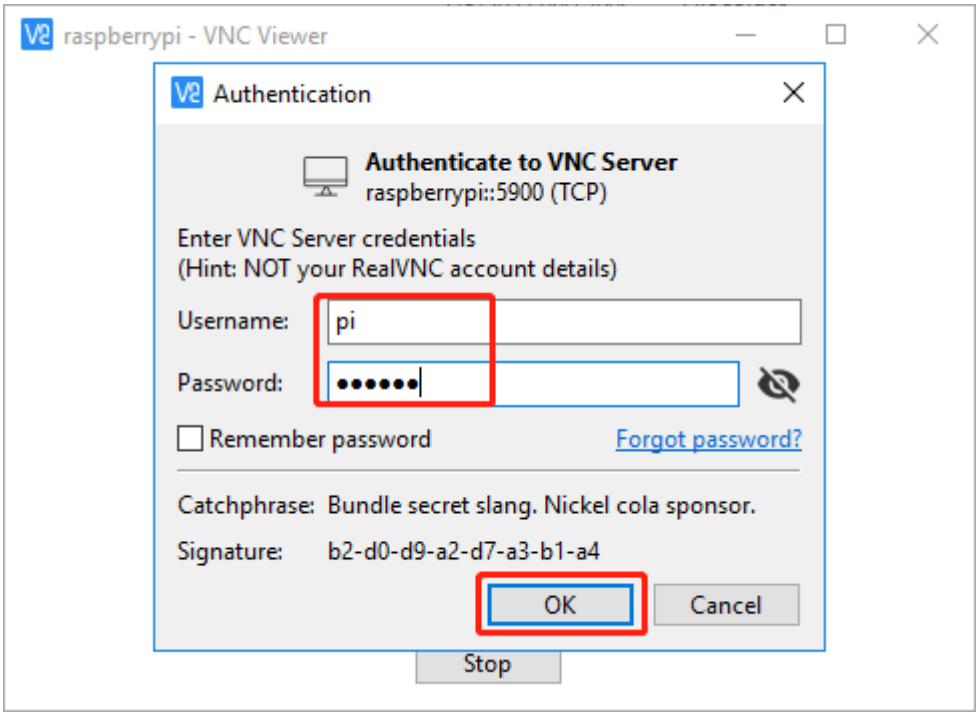


登录 VNC

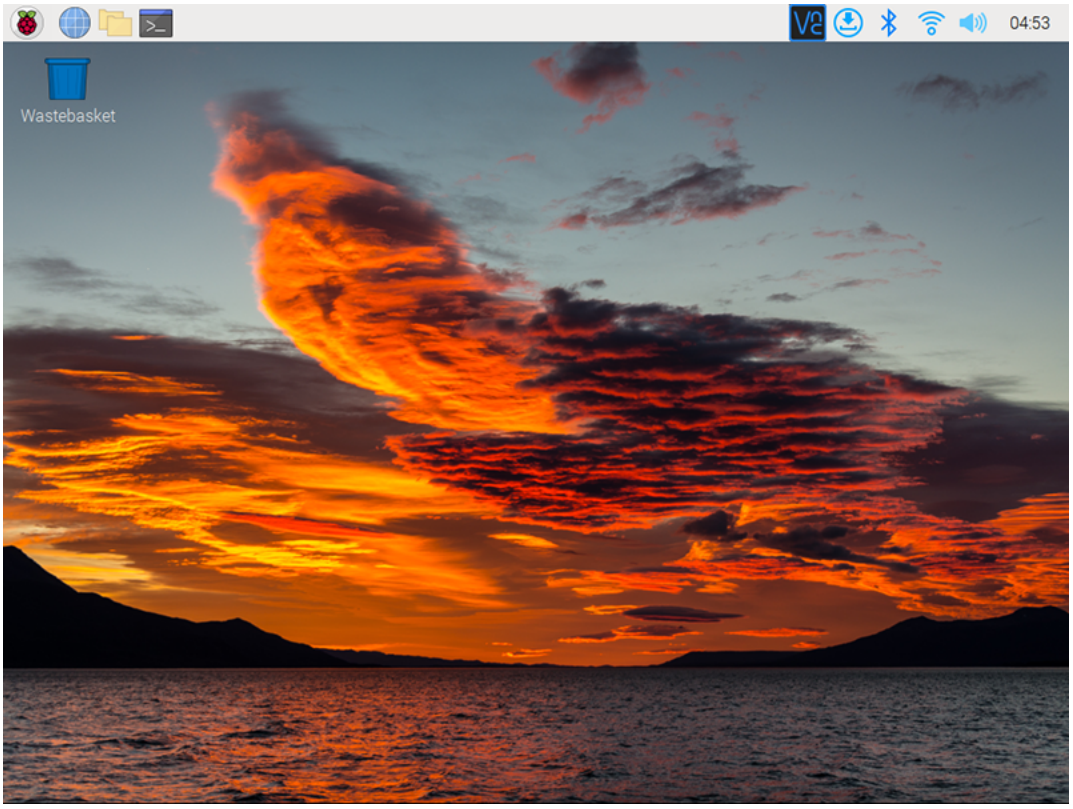
1. 你需要在个人电脑上下载并安装 VNC Viewer。
2. 安装完成后，打开它。然后，输入主机名或 IP 地址并按 Enter 键。



3. 在输入你的树莓派名字和密码后，点击 **OK**。

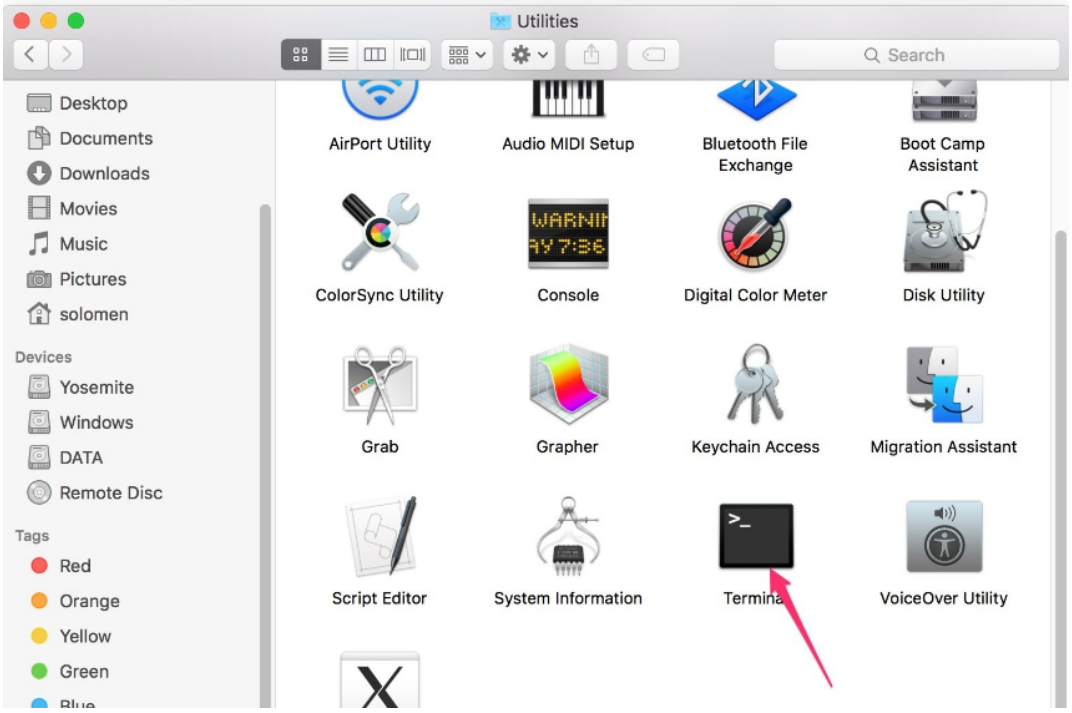


4. 现在你可以看到树莓派的桌面了。



Linux /Unix 用户

1. 跳转到 **应用程序-> 实用工具**，找到 **终端**，然后打开它。



2. 通过输入 `ping < 主机名>.local` 来检查你的 Raspberry Pi 是否在同一网络中。

```
ping raspberrypi.local
```

```
mac — pi@raspberrypi: ~ — -bash — 80×24
Last login: Wed Dec 21 10:20:41 on console
-bash: $: command not found

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
MacdeMBP:~ mac$ ping raspberrypi.local
PING raspberrypi.local (192.168.6.116): 56 data bytes
64 bytes from 192.168.6.116: icmp_seq=0 ttl=64 time=0.464 ms
64 bytes from 192.168.6.116: icmp_seq=1 ttl=64 time=0.315 ms
64 bytes from 192.168.6.116: icmp_seq=2 ttl=64 time=0.431 ms
64 bytes from 192.168.6.116: icmp_seq=3 ttl=64 time=0.328 ms
64 bytes from 192.168.6.116: icmp_seq=4 ttl=64 time=0.456 ms
64 bytes from 192.168.6.116: icmp_seq=5 ttl=64 time=0.355 ms
```

如上图所示，当 Raspberry Pi 连接到网络后，你可以看到它的 IP 地址。

- 如果终端提示 `Ping request could not find host pi.local. Please check the name and try again.`，请按照提示确保你输入的主机名是正确的。
- 仍然无法获得 IP？请检查 Raspberry Pi 上的网络或 WiFi 配置。

3. 输入 `ssh < 用户名>@< 主机名>.local` (或 `ssh < 用户名>@<IP 地址>`)。

```
ssh pi@raspberrypi.local
```

备注： 如果出现提示 `The term 'ssh' is not recognized as the name of a cmdlet...`。

这意味着你的系统太旧，没有预装 ssh 工具，你需要手动通过 *Powershell 安装 OpenSSH*。

或使用第三方工具如*PuTTY*。

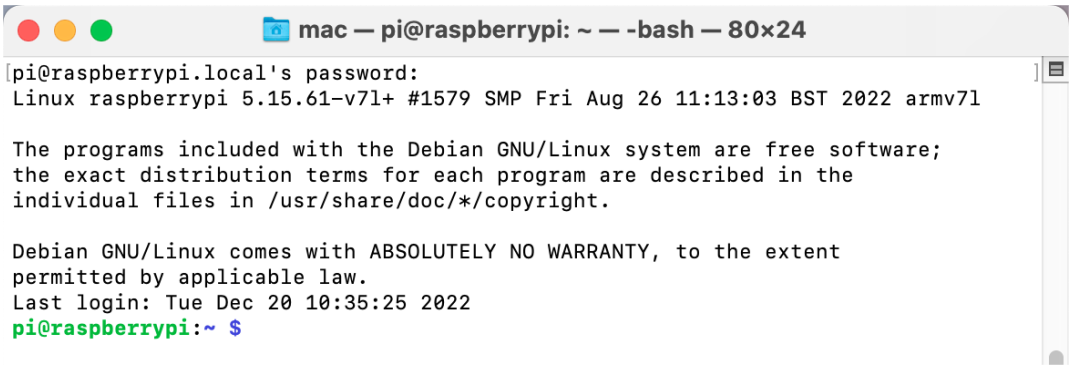
4. 下面的信息只会在你第一次登录时显示，所以输入 `yes`。

```
The authenticity of host 'raspberrypi.local␣
↪(2400:2410:2101:5800:635b:f0b6:2662:8cba)' can't be established.
ED25519 key fingerprint is SHA256:oo7x3ZSgAo032wD1tE8eW0fFM/
↪kmewIvRwkBys6XRwg.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

5. 输入你之前设置的密码。(我的是 `raspberry`。)

备注： 当你输入密码时，字符不会在窗口中显示，这是正常的。你需要做的是输入正确的密码。

6. 我们现在已经连接上 Raspberry Pi，可以进入下一步了。



5.1.4 安装所有模块（重要）

确保你已连接到互联网并更新你的系统:

```
sudo apt update
sudo apt upgrade
```

备注： 如果你安装的是 Lite 版本的操作系统，必须安装 Python3 相关的包。

```
sudo apt install git python3-pip python3-setuptools python3-smbus
```

安装 `robot-hat`。

```
cd ~/
git clone -b v2.0 https://github.com/sunfounder/robot-hat.git
```

(续下页)

(接上页)

```
cd robot-hat
sudo python3 setup.py install
```

接下来下载并安装 vilib 模块。

```
cd ~/
git clone -b picamera2 https://github.com/sunfounder/vilib.git
cd vilib
sudo python3 install.py
```

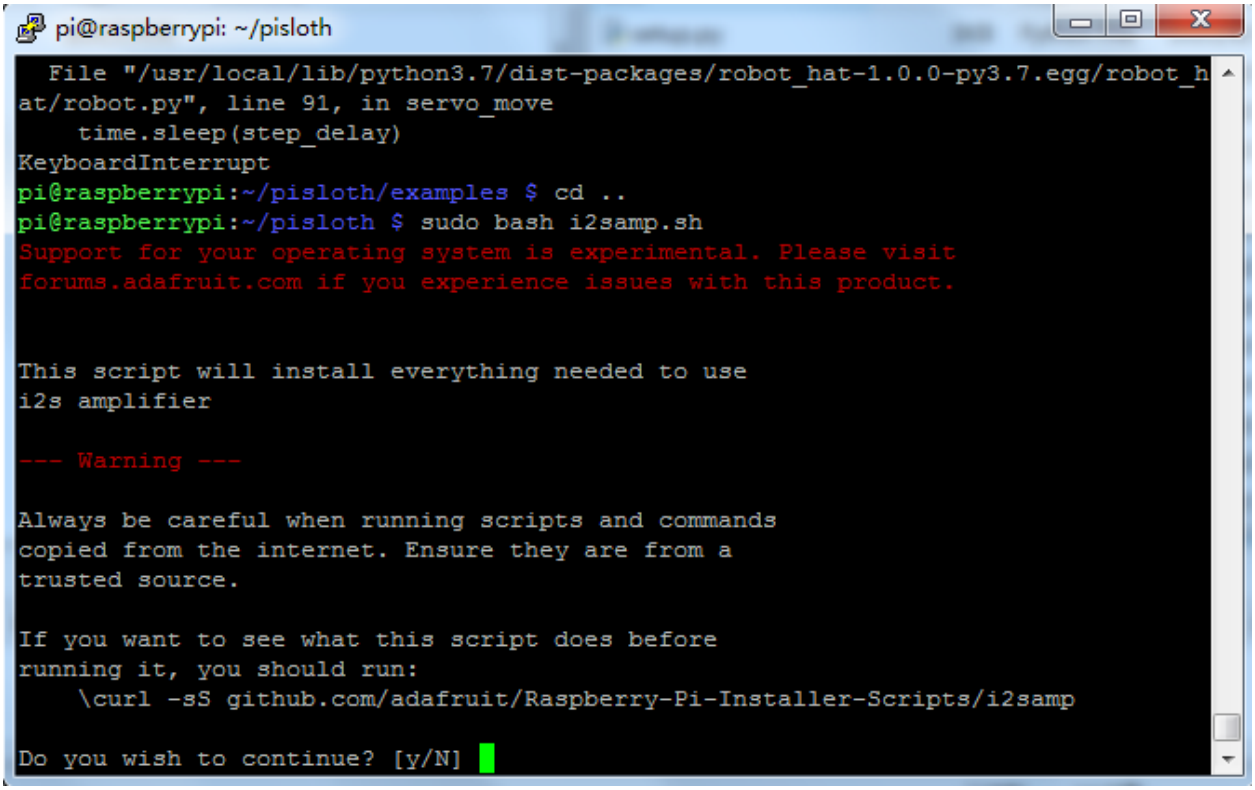
下载并安装 picar-x 模块。

```
cd ~/
git clone -b v2.0 https://github.com/sunfounder/picar-x.git
cd picar-x
sudo python3 setup.py install
```

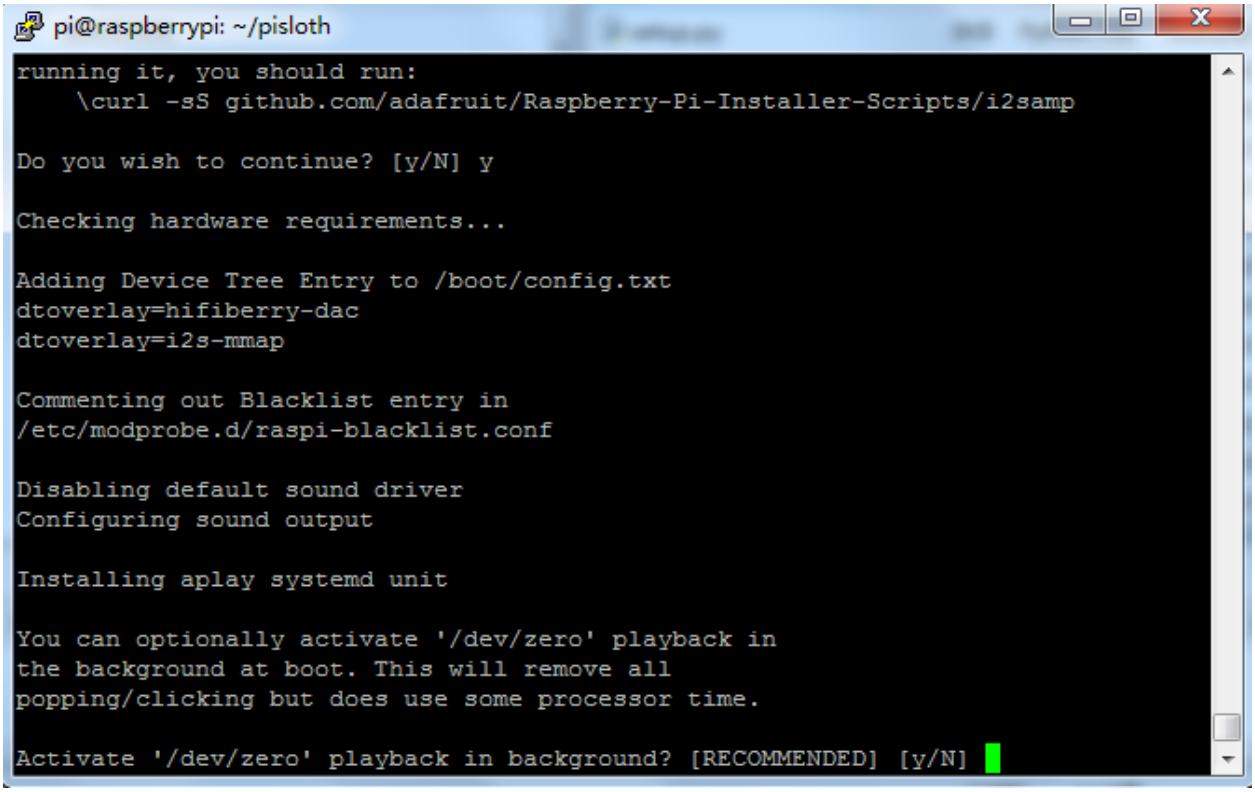
这一步需要一点时间，请耐心等待。

最后，你需要运行脚本 i2samp.sh 来安装 i2s 放大器所需的组件，否则 picar-x 将没有声音。

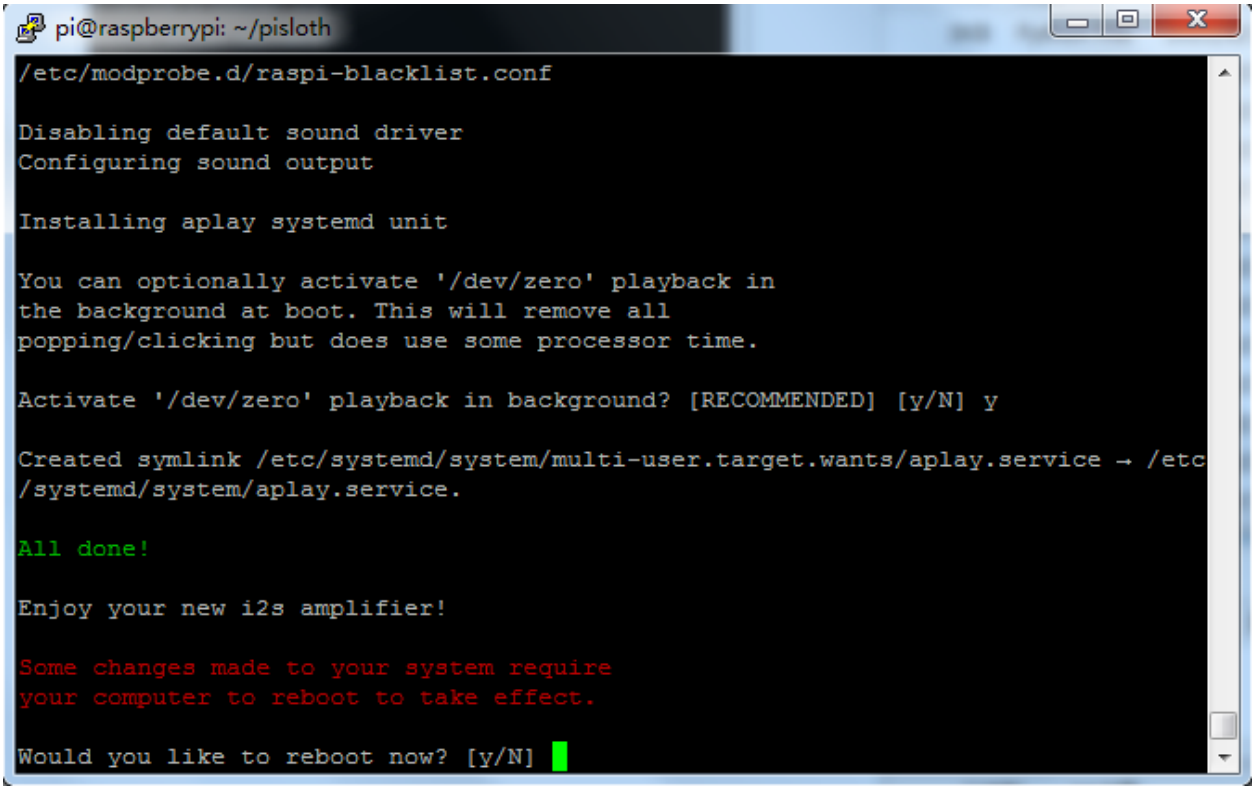
```
cd ~/picar-x
sudo bash i2samp.sh
```



输入 y 并按回车继续运行脚本。



输入 y 并按回车在后台运行 /dev/zero。



输入 y 并按回车重启 Picar-X。

备注： 如果重启后没有声音，你可能需要多次运行 i2samp.sh 脚本。

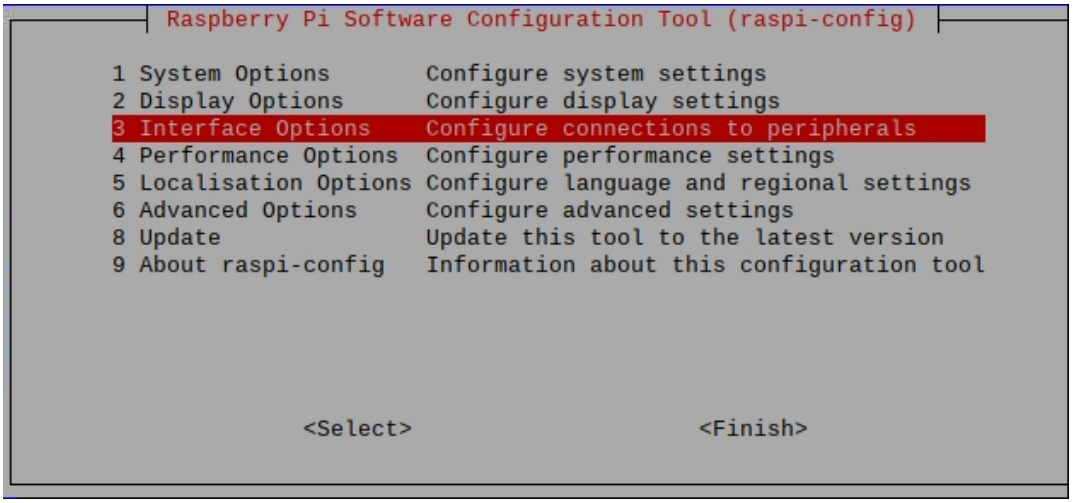
5.1.5 启用 I2C 接口（重要）

在这里，我们使用的是 Raspberry Pi 的 I2C 接口，但默认情况下它们是禁用的，所以我们首先需要启用它们。

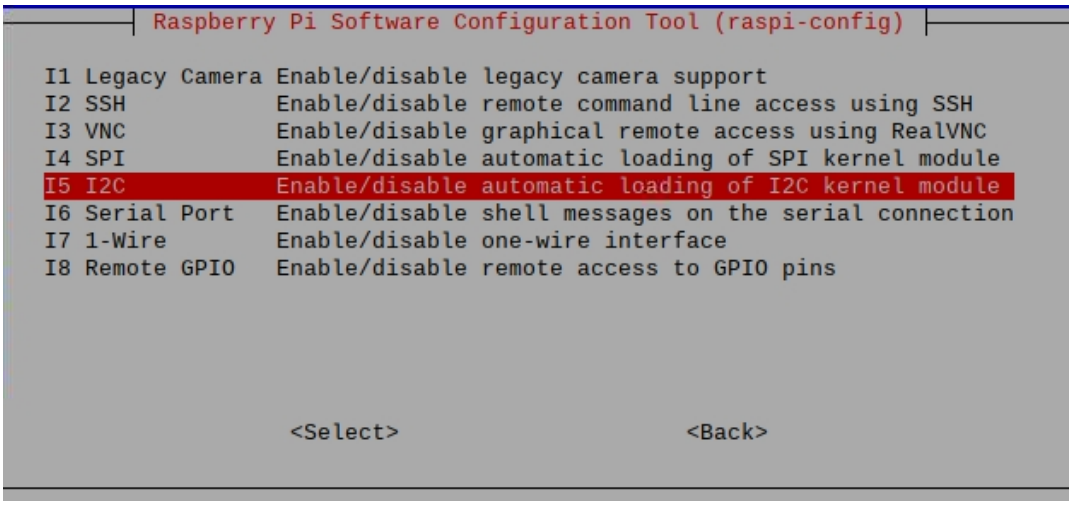
1. 输入以下命令：

```
sudo raspi-config
```

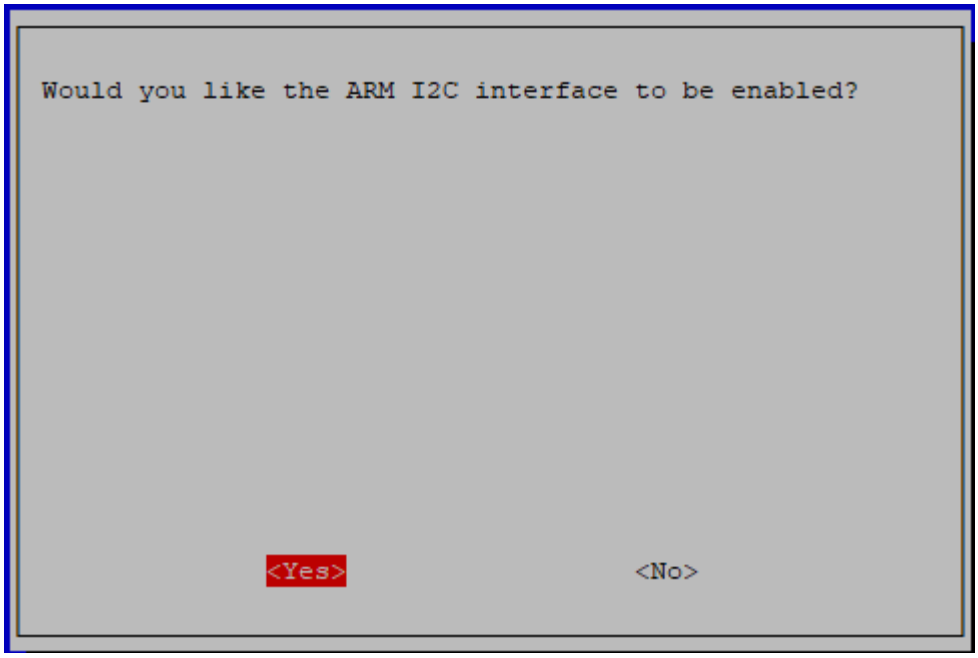
2. 使用键盘的向下箭头键选择 **Interfacing Options**，然后按 **Enter** 键。



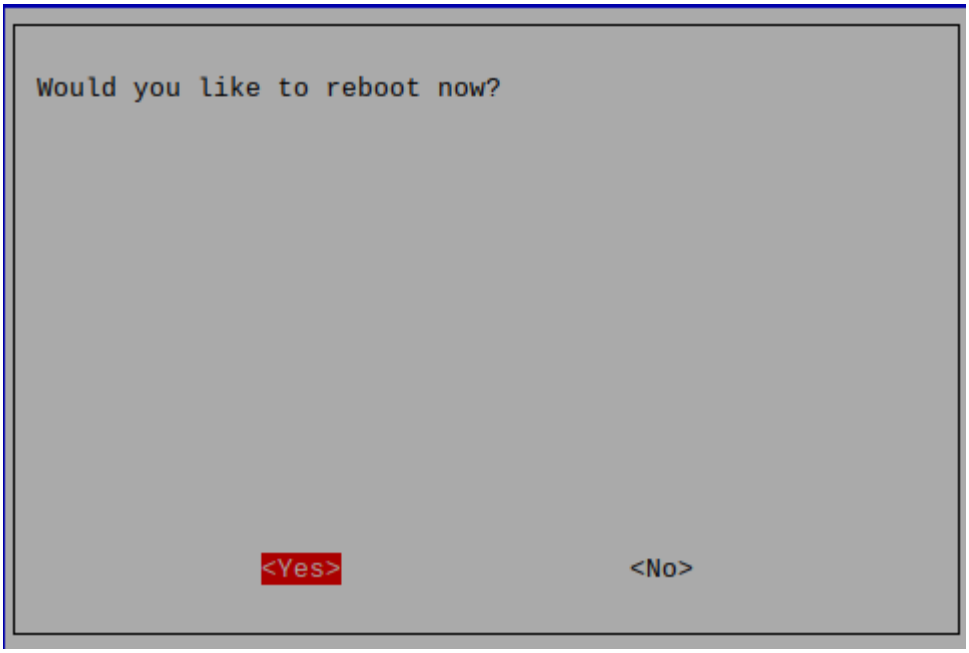
3. 接着选择 **I2C**。



4. 使用键盘上的箭头键选择 **<Yes>** -> **<OK>** 以完成 I2C 的设置。



5. 当你选择 **<Finish>** 后，会弹出一个提醒告诉你需要重启以使设置生效，选择 **<Yes>**。

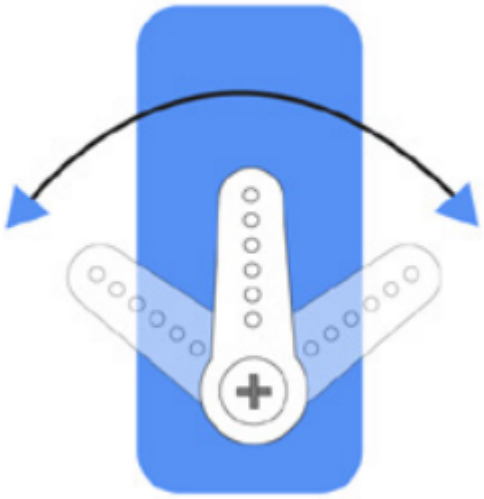


5.1.6 伺服机调整 (重要)

伺服机的角度范围是-90~90 度，但出厂时设置的角度是随机的，可能是 0°，也可能是 45°；如果我们直接以这样的角度组装它，它将在机器人运行代码后导致混乱的状态，或者更糟，它将导致伺服机堵塞和烧毁。

所以在这里我们需要将所有的伺服机角度设置为 0°，然后再安装它们，这样伺服机的角度就在中间，无论向哪个方向转。

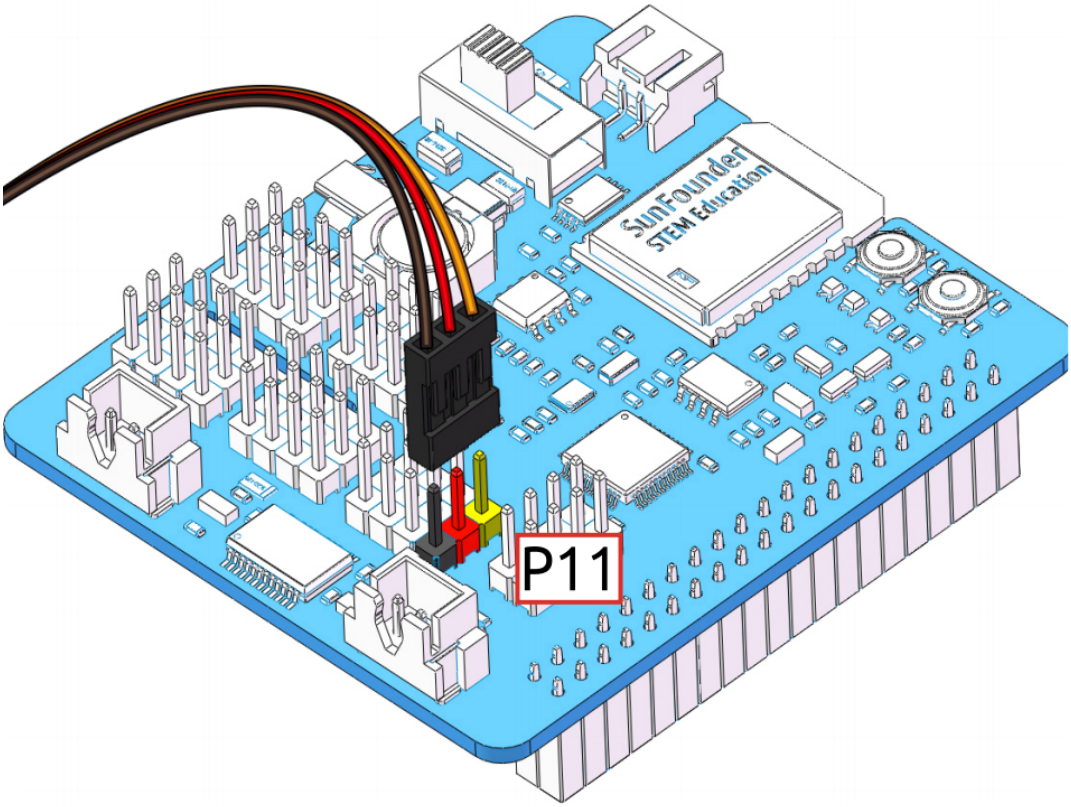
1. 为确保伺服机已正确设置为 0°，首先将伺服机臂插入伺服机轴，然后轻轻旋转摇臂到不同的角度。这个伺服机臂只是为了让你清楚地看到伺服机正在旋转。



2. 现在，在 `example/` 文件夹中运行 `servo_zeroing.py`。

```
cd ~/picar-x/example
sudo python3 servo_zeroing.py
```

3. 接下来，将伺服机电缆插入 P11 端口，如下所示，同时你会看到伺服机臂旋转到一个位置 (这是 0° 的位置，这是一个随机的位置，可能不是垂直或平行的)。



4. 现在，拔下伺服机臂，确保伺服线仍然连接，不要关闭电源。然后按照纸上的指示继续组装。

备注:

- 在用伺服螺丝固定这个伺服之前，不要拔下这个伺服电缆，可以在固定后拔下。
- 不要在上电状态下转动伺服机，以免损坏；如果伺服轴插入角度错误，拔出伺服机并重新插入。
- 在组装每个伺服机之前，需要将伺服电缆插入 P11 并打开电源，将其角度设置为 0°。

帕克组装完成后，尝试运行下面的项目。

5.2 校准 PiCar-X

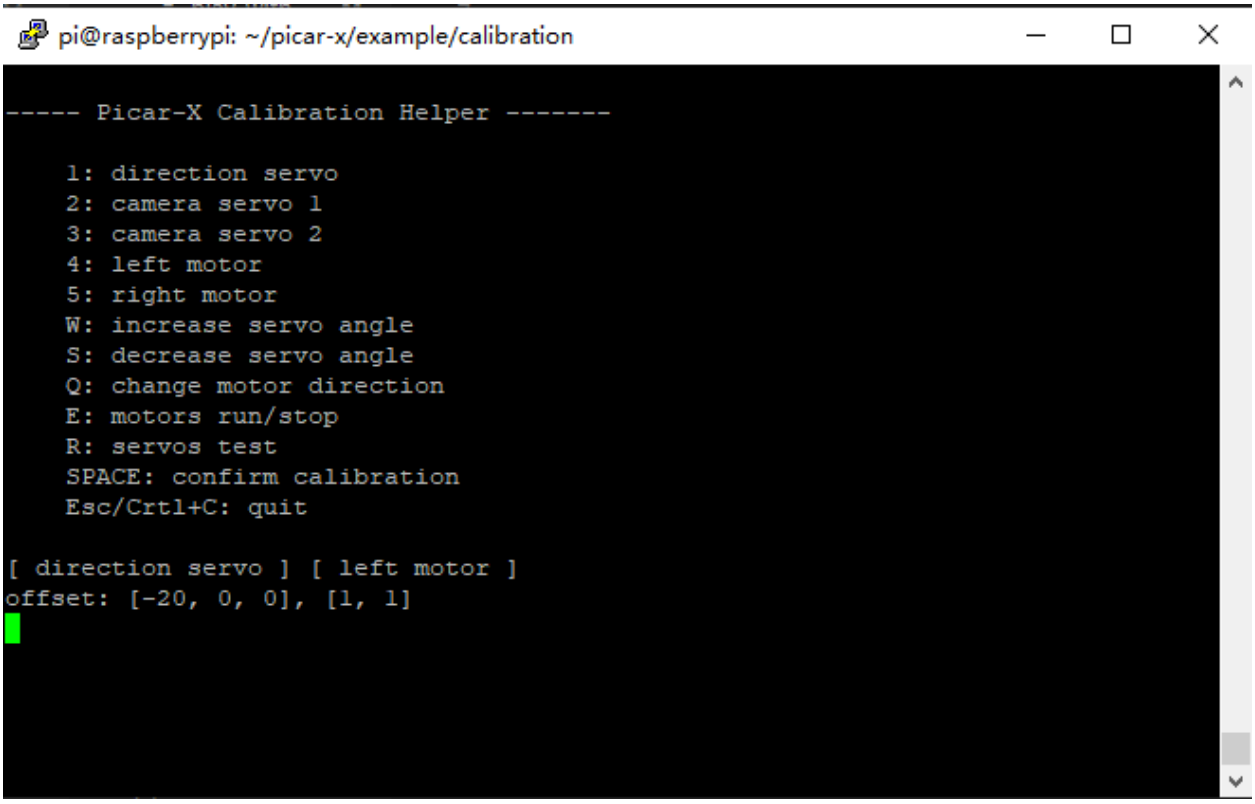
由于 PiCar-X 安装过程中可能出现的偏差或舵机本身的限制，某些舵机角度可能会略微倾斜，因此您可以对其进行校准。

当然，如果您认为组装是完美的并且不需要校准，您可以跳过这一章。

运行校准程序 `calibration.py`

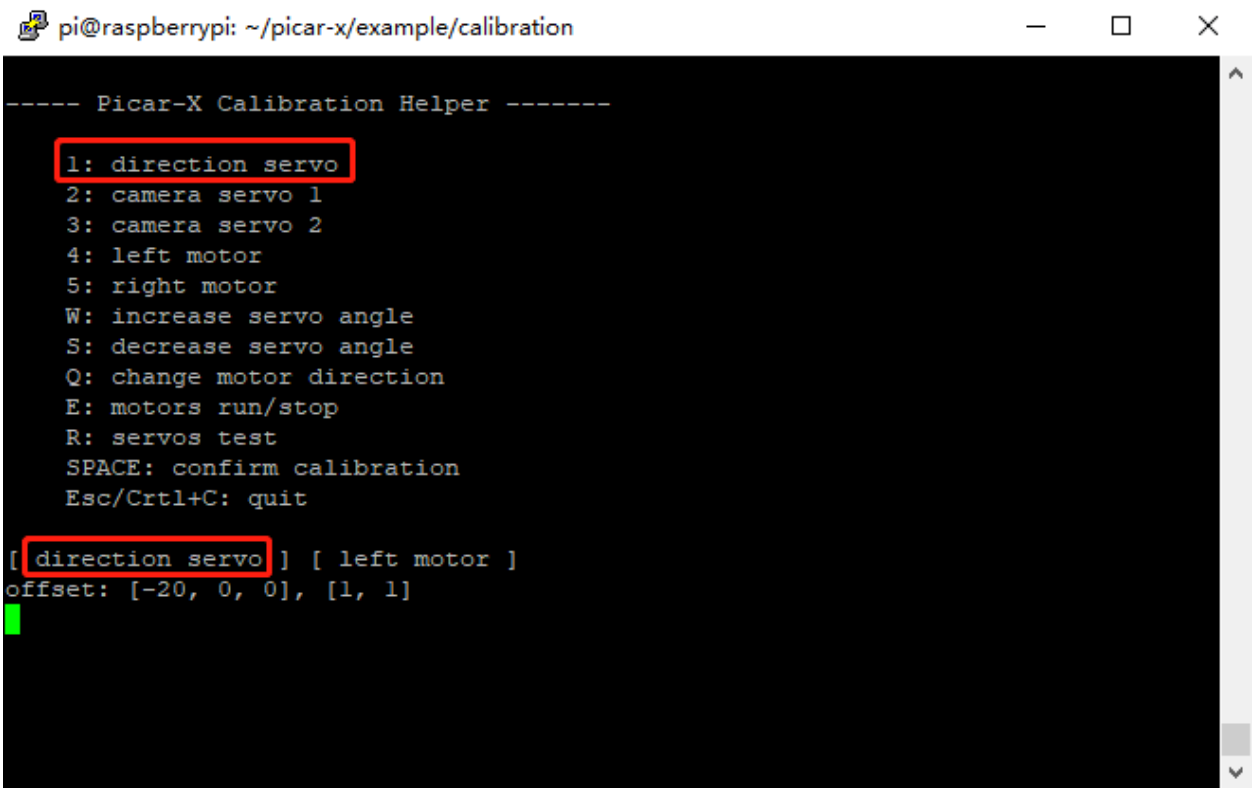
```
cd /home/pi/picar-x/examples/calibration
sudo python3 calibration.py
```

运行上述代码后，您会看到终端显示如下界面。

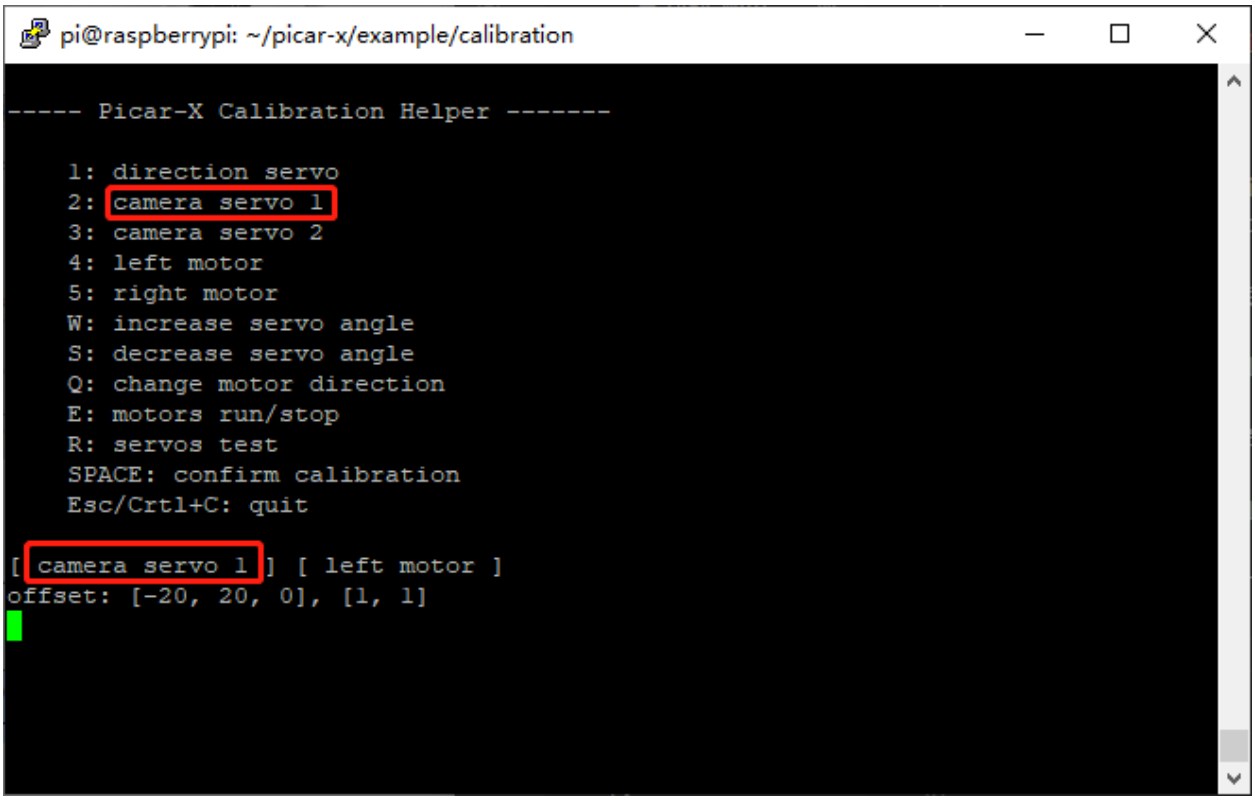


R 键用于测试控制前轮方向的舵机是否能正常工作，有没有损坏。

按数字键 1 可以选择前轮舵机，然后按 W/S 键可以慢慢地校准前轮的方向，让其看起来尽可能正向前方，而不左右偏斜。

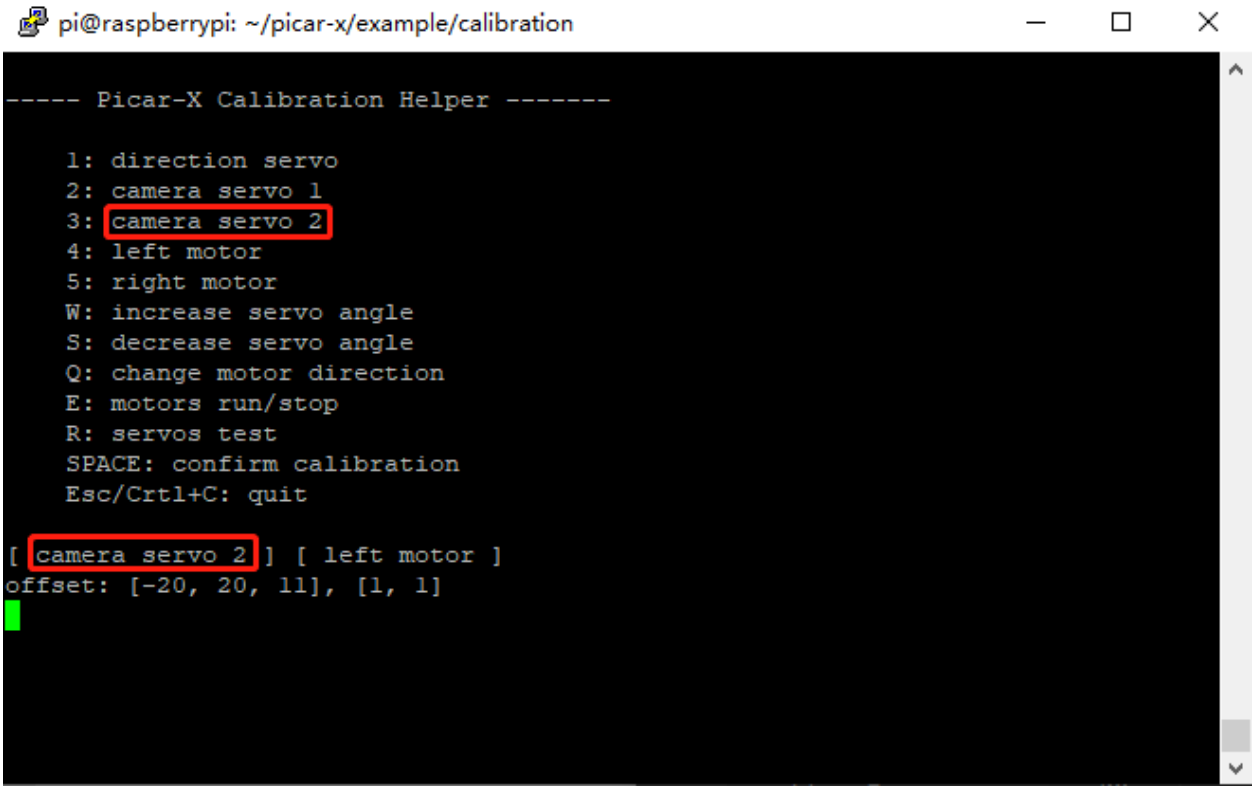


按数字键 2 可以选择摄像头左右方向的舵机，然后按 W/S 键可以慢慢地校准摄像头左右的方向，让其正视前方不左右倾斜。

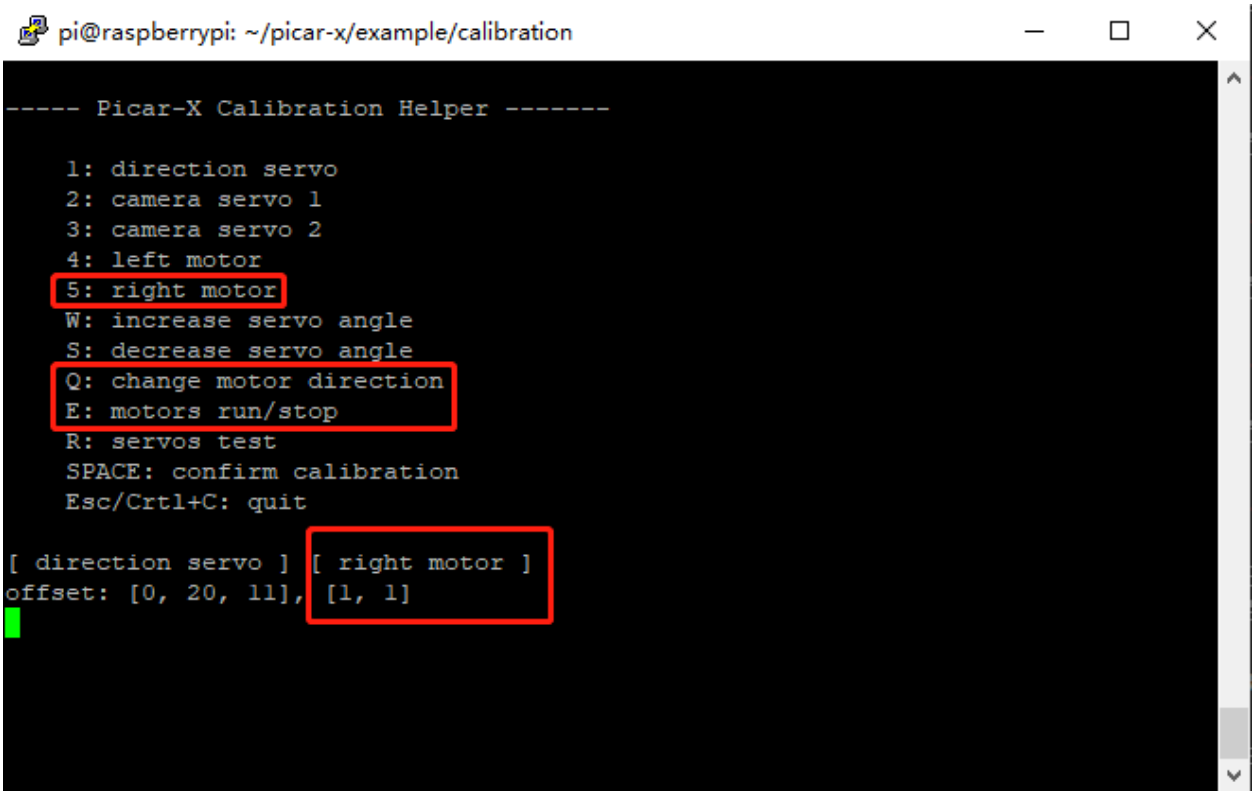


按数字键 3 可以选择摄像头上下方向的舵机，然后按 W/S 键可以慢慢地校准摄像头上下的方向，让其平视

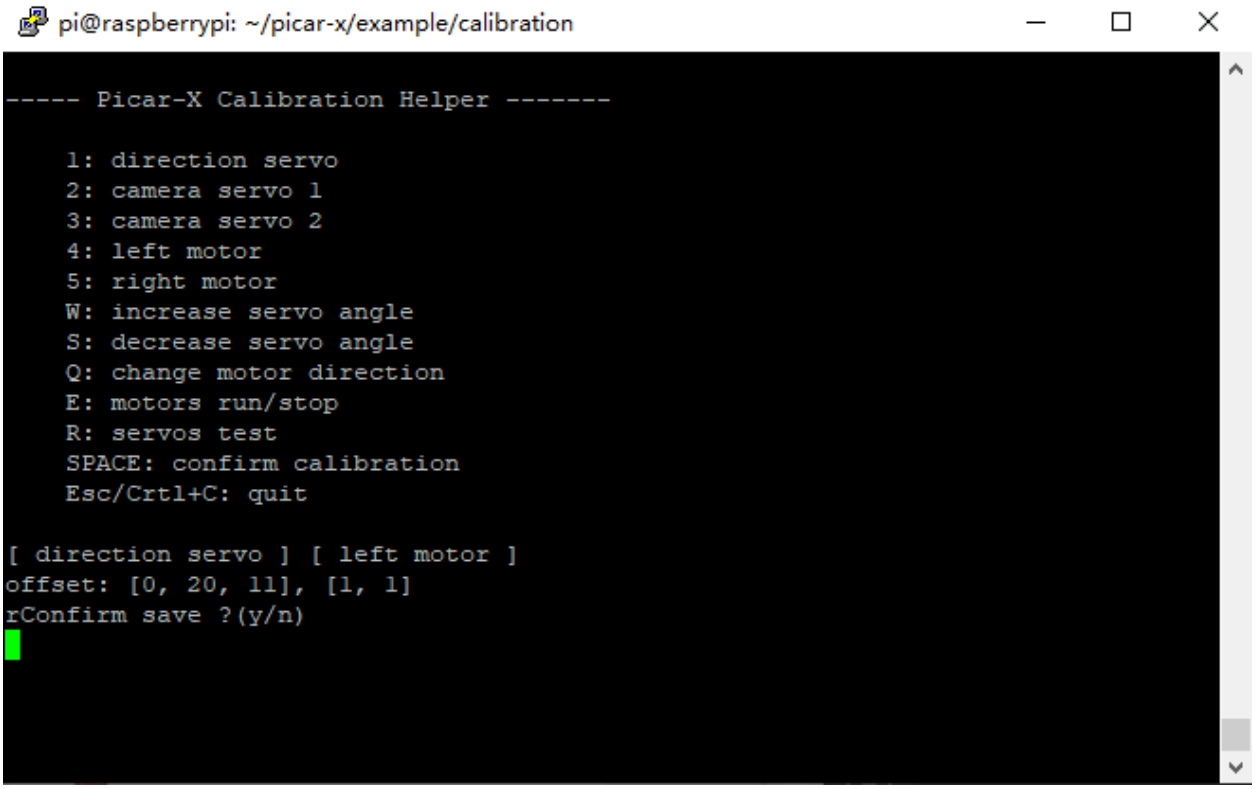
前方不上下倾斜。



由于安装时可能失误将后轮电机的线接反，导致其不能正常前进，可以按 E 测试小车是否能正常前进。若后轮的旋转方向反了，则可以按数字键 4 和 5 分别选择后轮左右两个马达，然后按 Q 键来校准其旋转方向。



当校准完毕之后按下空格键保存校准参数，会有提示输入 y 确认，然后按 esc 退出程序完成校准。



5.3 让帕克动起来

这是第一个项目，让我们测试一下帕克的基本运动。

运行代码

```
cd /home/pi/picar-x/example
sudo python3 move.py
```

运行代码后，帕克会向前移动，S 形转弯，停下来摇头。

代码

```
from picarx import Picarx
import time

if __name__ == "__main__":
    try:
        px = Picarx()
        px.forward(30)
        time.sleep(0.5)
        for angle in range(0,35):
            px.set_dir_servo_angle(angle)
            time.sleep(0.01)
        for angle in range(35,-35,-1):
            px.set_dir_servo_angle(angle)
```

(续下页)

(接上页)

```
            time.sleep(0.01)
        for angle in range(-35,0):
            px.set_dir_servo_angle(angle)
            time.sleep(0.01)
        px.forward(0)
        time.sleep(1)

        for angle in range(0,35):
            px.set_camera_servo1_angle(angle)
            time.sleep(0.01)
        for angle in range(35,-35,-1):
            px.set_camera_servo1_angle(angle)
            time.sleep(0.01)
        for angle in range(-35,0):
            px.set_camera_servo1_angle(angle)
            time.sleep(0.01)
        for angle in range(0,35):
            px.set_camera_servo2_angle(angle)
            time.sleep(0.01)
        for angle in range(35,-35,-1):
            px.set_camera_servo2_angle(angle)
            time.sleep(0.01)
        for angle in range(-35,0):
            px.set_camera_servo2_angle(angle)
            time.sleep(0.01)

    finally:
        px.forward(0)
```

这个怎么运作？

帕克的基本功能在 `picarx` 模块中，可以用来控制舵机和车轮，并使帕克向前移动、S 形转弯或摇头。

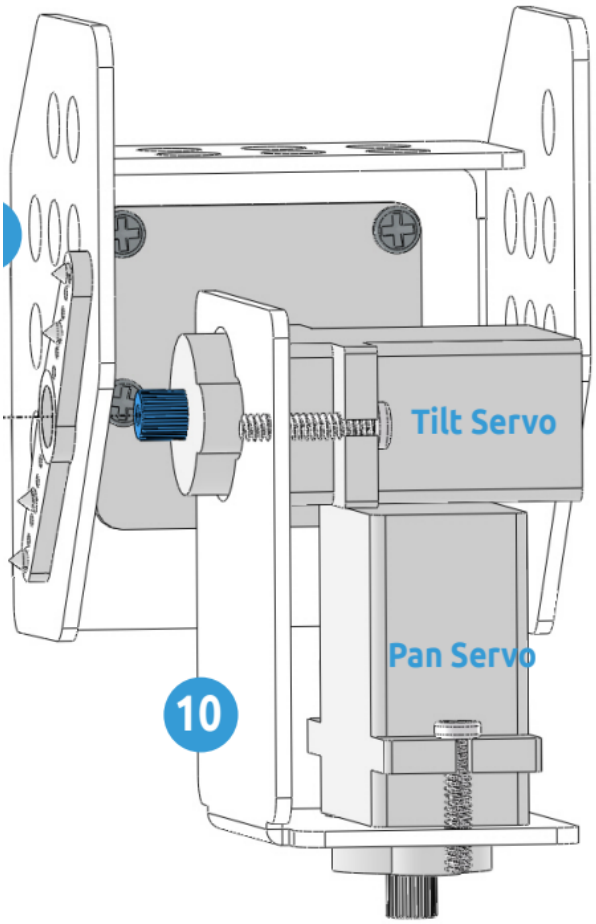
现在，导入了支持帕克基本功能的库。这些线条将出现在所有涉及帕克运动的示例中。

```
from picarx import Picarx
import time
```

然后使用带有 `for` 循环的以下函数使帕克向前移动、改变方向和移动相机的平移/倾斜。

```
px.forward(speed)
px.set_dir_servo_angle(angle)
px.set_camera_servo1_angle(angle)
px.set_camera_servo2_angle(angle)
```

- `forward()`：命令帕克以给定的速度前进。
- `set_dir_servo_angle()`：将转向舵机转向特定的角度。
- `set_camera_servo1_angle()`：将平移伺服器转到特定的角度。
- `set_camera_servo2_angle()`：将倾斜伺服转向特定的角度。



5.4 避障

在这个项目中，帕克会在前进的同时检测前方的障碍物，当障碍物太近时，它会改变前进的方向。

运行代码

```
cd /home/pi/picar-x/example
sudo python3 avoiding_obstacles.py
```

运行代码后，帕克会向前走。

如果检测到前方障碍物的距离小于 25cm，则向左转。

若左转后方向无障碍物或障碍物距离大于 25cm，则继续前进。

代码

```
from picarx import Picarx


def main():
    try:
        px = Picarx()
        # px = Picarx(ultrasonic_pins=['D2','D3']) # tring, echo
```

(续下页)

(接上页)

```
        px.forward(30)
        while True:
            distance = px.ultrasonic.read()
            print("distance: ",distance)
            if distance > 0 and distance < 300:
                if distance < 25:
                    px.set_dir_servo_angle(-35)
                else:
                    px.set_dir_servo_angle(0)
    finally:
        px.forward(0)


if __name__ == "__main__":
    main()
```

这个怎么运作？

picarx 模块中还导入了 ultrasonic 模块，我们可以用其封装的一些功能来检测距离。

```
from picarx import Picarx
```

因为 picarx 模块中导入了 ultrasonic 模块，我们可以直接用 `px.ultrasonic.read()` 来获取距离。

```
px = Picarx()
px.forward(30)
while True:
    distance = px.ultrasonic.read()
```

下面的代码片段读取超声波模块报告的距离值，如果距离低于 25 厘米（10 英寸），它将把转向伺服从 0°（直线）设置为 -35°（左转）。

```
while True:
    distance = px.ultrasonic.read()
    print("distance: ",distance)
    if distance > 0 and distance < 300:
        if distance < 25:
            px.set_dir_servo_angle(-35)
        else:
            px.set_dir_servo_angle(0)
```

5.5 线路跟踪

本项目将使用灰度模块使帕克沿线向前移动。使用深色胶带使线条尽可能平滑，转弯幅度不要太大，避免帕克脱轨。

运行代码

```
cd /home/pi/picar-x/example
sudo python3 minecart_plus.py
```

运行代码后，帕克会沿着线向前移动。

代码

```
from picarx import Picarx

if __name__=='__main__':
try:
    px = Picarx()
    px_power = 10
    while True:
        gm_val_list = px.get_grayscale_data()
        print("gm_val_list:",gm_val_list)
        gm_status = px.get_line_status(gm_val_list)
        print("gm_status:",gm_status)

        if gm_status == 'forward':
            print(1)
            px.forward(px_power)

        elif gm_status == 'left':
            px.set_dir_servo_angle(12)
            px.forward(px_power)

        elif gm_status == 'right':
            px.set_dir_servo_angle(-12)
            px.forward(px_power)
        else:
            px.set_dir_servo_angle(0)
            px.stop()
finally:
    px.stop()
```

这个怎么运作？

picarx 模块中也导入了灰度传感器模块 grayscale_module ，我们可以用其中的一些方法来检测黑线。

检测黑线的函数如下所示：

- `get_grayscale_data()`：该函数直接输出三个传感器的读数，从右到左。区域越亮，获得的值越大。
- `get_line_status()`：该函数将根据三个探测器检测到的值生成一个动作。有四种类型的动作：forward 、left 、right 和 stop 。

这些动作的触发条件如下：模块中默认分配了一个数值作为检测到黑色还是白色的阈值。当三个探针的检测值均大于阈值时，表示探头感应到的是白色，没有检测到则为黑线，检测到黑线会使 `get_line_status()` 返回一个 `stop` 参数。

- 如果右侧（也是第一个）探针检测到黑线，则返回 `right` 。
- 如果中间探针检测到黑线，则返回 `forward`；
- 如果左探针检测到黑线，则返回 `left` 。

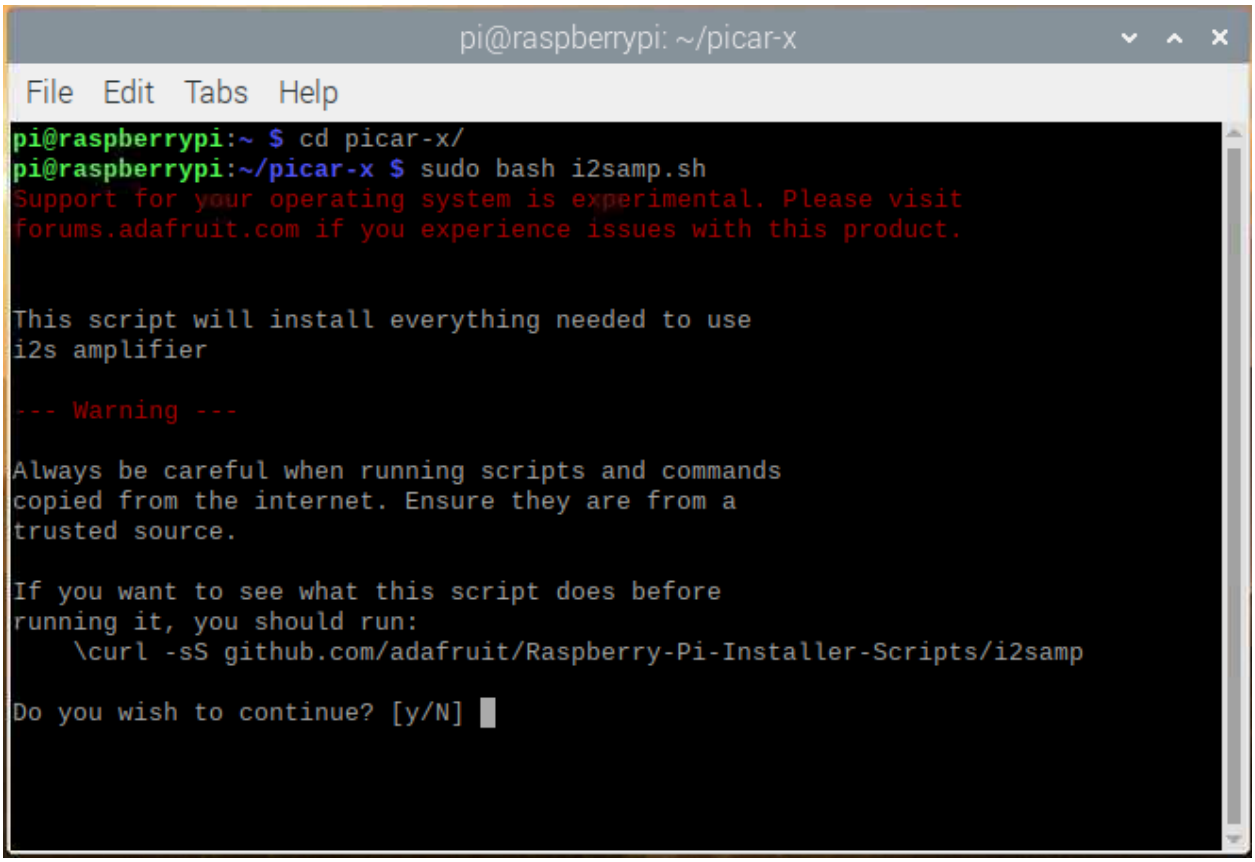
5.6 文字转语音

在使用文字转语音 (TTS) 功能之前，请先激活扬声器，使其启用并可以发出声音。

在 **picar-x** 文件夹中运行 `i2samp.sh`，此脚本将安装使用 i2s 放大器所需的一切。

运行代码

```
cd /home/pi/picar-x
sudo bash i2samp.sh
```



会有几个提示要求确认请求。用 **Y** 响应所有提示。对树莓派系统进行更改后，需要重新启动计算机才能使这些更改生效。

重新启动后，再次运行 `i2samp.sh` 脚本来测试放大器。如果扬声器成功播放声音，则配置完成。

运行代码

```
cd /home/pi/picar-x/example
sudo python3 tts_example.py
```

运行代码后，帕克会说“你好”、“嗨”、“再见”、“很高兴见到你”。

代码

```
from robot_hat import TTS


if __name__ == "__main__":
```

(续下页)

(接上页)

```
    words = ["Hello", "Hi", "Good bye", "Nice to meet you"]
    tts_robot = TTS()
    for i in words:
        print(i)
        tts_robot.say(i)
```

这个怎么运作？

eSpeak 软件用于实现 TTS 的功能。

导入 robot_hat 中的 TTS 模块，其中封装了可以将文字转换成语音的函数。

```
from robot_hat import TTS
```

创建一个字符串列表 `words`，然后创建 TTS() 类的实例化对象 `tts_robot`，最后用 `tts_robot.say()` 函数将列表中的文字用语音说出来。

```
words = ["Hello", "Hi", "Good bye", "Nice to meet you"]
tts_robot = TTS()
for i in words:
    print(i)
    tts_robot.say(i)
```

5.7 计算机视觉

这个下一个项目将正式进入计算机视觉领域！

要执行接下来的四个实验，请确保已完成远程桌面。通过 SSH 的远程连接不会显示摄像机图像。

运行代码

备注:

- 这个项目需要访问树莓派的桌面来查看相机模块拍摄的画面。
- 你可以将屏幕连接到 PiCar-X 上，或者参考教程远程桌面，用 VNC 或 XRDP 访问它。
- 一旦进入树莓派的桌面，打开 Terminal 并输入以下命令来运行它，或者直接用 Python 编辑器打开并运行它。

```
cd /home/pi/picar-x/example
sudo python3 computer_vision.py
```

代码运行后，你将在窗口上看到摄像头模块拍摄的画面。

代码

```
import cv2
from picamera.array import PiRGBArray
from picamera import PiCamera
import time


with PiCamera() as camera:
```

(续下页)

(接上页)

```
    camera.resolution = (640, 480)
    camera.framerate = 24
    rawCapture = PiRGBArray(camera, size=camera.resolution)
    time.sleep(2)

    for frame in camera.capture_continuous(rawCapture, format="bgr",use_video_
↪port=True): # use_video_port=True
        img = frame.array
        cv2.imshow("video", img)  # OpenCV image show
        rawCapture.truncate(0)  # Release cache

        k = cv2.waitKey(1) & 0xFF
        if k == 27:
            break

    print('quit ...')
    cv2.destroyAllWindows()
    camera.close()
```

这个怎么运作？

照片是用 `PiCamera` 获得的。这个包为树莓派相机提供了一个纯 Python 接口。

- PiCamera Docs

将图像捕获到文件只需要将文件名指定为所需的任何 `capture()` 方法的输出。

```
from time import sleep
from picamera import PiCamera

with PiCamera() as camera:
    camera.resolution = (640, 480)
    camera.start_preview()
    # Camera warm-up time
    sleep(2)
    camera.capture('foo.jpg')
```

该项目使用 **capturing timelapse sequences** 函数。这种方法使 OpenCV 能够获取连续帧。

使用这种方法，相机会不断捕捉图像，直到被告知停止。图像会被自动赋予唯一的名称。`sleep(x)` 函数控制捕获之间的延迟。

```
from time import sleep
from picamera import PiCamera

with PiCamera() as camera:
    camera.resolution = (640, 480)
    camera.start_preview()
    sleep(2)

    for filename in camera.capture_continuous('img{counter:03d}.jpg'):
        print('Captured %s' % filename)
        sleep(10) #  capture images with a 10s delay between each shot
```

为了捕获 OpenCV 对象，图像将被捕获到 Python 的内存流类：`BytesIO`。BytesIO 会将流转换为 `numpy` 数组，程序将使用 OpenCV 读取该数组：

- What is Numpy?

```
import io
import time
import picamera
import cv2
import numpy as np

# Create the in-memory stream
stream = io.BytesIO()
with picamera.PiCamera() as camera:
    camera.start_preview()
    time.sleep(2)
    camera.capture(stream, format='jpeg')
# Construct a numpy array from the stream
data = np.fromstring(stream.getvalue(), dtype=np.uint8)
# "Decode" the image from the array, preserving colour
image = cv2.imdecode(data, 1)
# OpenCV returns an array with data in BGR order. If you want RGB instead
# use the following...
image = image[:, :, ::-1]
```

为了避免 JPEG 编码和解码的损失，请使用 picamera.array 模块中的类。这也有可能提高图像处理的速度。

由于 OpenCV 图像只是按 BGR 顺序排列的 `numpy` 数组，`PiRGBArray` 类，并且简单地使用 `bgr` 格式捕获。注：RGB 数据和 BGR 数据大小相同，配置相同，但颜色平面相反。

- PiRGBArray

```
import time
import picamera
import picamera.array
import cv2

with picamera.PiCamera() as camera:
    camera.start_preview()
    time.sleep(2)
    with picamera.array.PiRGBArray(camera) as stream:
        camera.capture(stream, format='bgr')
        # At this point the image is available as stream.array
        image = stream.array
```

结合捕捉延时序列的方法，这些 3 维 RGB 数组由 OpenCV 展示。

```
import cv2
from picamera.array import PiRGBArray
from picamera import PiCamera

#init camera
with PiCamera() as camera:
    camera.resolution = (640,480)
    camera.framerate = 24
    rawCapture = PiRGBArray(camera, size=camera.resolution)

    for frame in camera.capture_continuous(rawCapture, format="bgr",use_video_
↪port=True): # use_video_port=True
        img = frame.array
        cv2.imshow("video", img)  # OpenCV image show
        rawCapture.truncate(0)  # Release cache
```

(续下页)

(接上页)

```
        # click ESC key to exit.
        k = cv2.waitKey(1) & 0xFF
        if k == 27:
            camera.close()
            break
```

还有许多其他方法可以使用 OpenCV 读取视频流。这些示例中使用的那些更适合接下来的四个 PiCar-X 任务，例如颜色检测 和人脸检测。

更多视频流使用方式请参考：OpenCV-Python 教程 。

5.8 颜色检测

该项目将在之前的计算机视觉 项目中添加颜色检测算法。

- [PDF] 颜色卡

备注： 由于打印机碳粉或打印介质（如棕褐色纸）的差异，打印的颜色可能与 Python 颜色模型的色调略有不同。这会导致不太准确的颜色识别。

运行代码

备注：

- 这个项目需要访问树莓派的桌面来查看相机模块拍摄的画面。

- 你可以将屏幕连接到 PiCar-X 上，或者参考教程远程桌面，用 VNC 或 XRDP 访问它。
- 一旦进入树莓派的桌面，打开 Terminal 并输入以下命令来运行它，或者直接用 Python 编辑器打开并运行它。

```
cd /home/pi/picar-x/example
sudo python3 color_detect.py
```

当代码运行时，如果 PiCar-X 捕捉到一个红色物体，它会将其框出来。您还可以 `'red'` 将代码中的更改为另一种颜色以进行检测。

代码

```
import cv2
from picamera.array import PiRGBArray
from picamera import PiCamera
import numpy as np
import time

color_dict = {'red':[0,4],'orange':[5,18],'yellow':[22,37],'green':[42,85],'blue':[92,
↪110],'purple':[115,165],'red_2':[165,180]}  #Here is the range of H in the HSV␣
↪color space represented by the color

kernel_5 = np.ones((5,5),np.uint8) #Define a 5×5 convolution kernel with element␣
↪values of all 1.

def color_detect(img,color_name):

    # The blue range will be different under different lighting conditions and can be␣
↪adjusted flexibly.  H: chroma, S: saturation v: lightness
    resize_img = cv2.resize(img, (160,120), interpolation=cv2.INTER_LINEAR)  # In␣
↪order to reduce the amount of calculation, the size of the picture is reduced to␣
↪(160,120)
    hsv = cv2.cvtColor(resize_img, cv2.COLOR_BGR2HSV)              # Convert from BGR␣
↪to HSV
    color_type = color_name

    mask = cv2.inRange(hsv,np.array([min(color_dict[color_type]), 60, 60]), np.
↪array([max(color_dict[color_type]), 255, 255]) )           # inRange(): Make the␣
↪ones between lower/upper white, and the rest black
    if color_type == 'red':
            mask_2 = cv2.inRange(hsv, (color_dict['red_2'][0],0,0), (color_dict['red_2
↪'][1],255,255))
            mask = cv2.bitwise_or(mask, mask_2)

    morphologyEx_img = cv2.morphologyEx(mask, cv2.MORPH_OPEN, kernel_5,iterations=1) ␣
↪            # Perform an open operation on the image

    # Find the contour in morphologyEx_img, and the contours are arranged according␣
↪to the area from small to large.
    _tuple = cv2.findContours(morphologyEx_img,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_
↪SIMPLE)
    # compatible with opencv3.x and openc4.x
    if len(_tuple) == 3:
        _, contours, hierarchy = _tuple
    else:
        contours, hierarchy = _tuple
```

(续下页)

(接上页)

```
    color_area_num = len(contours) # Count the number of contours

    if color_area_num > 0:
        for i in contours:    # Traverse all contours
            x,y,w,h = cv2.boundingRect(i)      # Decompose the contour into the␣
↪coordinates of the upper left corner and the width and height of the recognition␣
↪object

            # Draw a rectangle on the image (picture, upper left corner coordinate,␣
↪lower right corner coordinate, color, line width)
            if w >= 8 and h >= 8: # Because the picture is reduced to a quarter of␣
↪the original size, if you want to draw a rectangle on the original picture to␣
↪circle the target, you have to multiply x, y, w, h by 4.
                x = x * 4
                y = y * 4
                w = w * 4
                h = h * 4
                cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),2)  # Draw a rectangular␣
↪frame
                cv2.putText(img,color_type,(x,y), cv2.FONT_HERSHEY_SIMPLEX, 1,(0,0,
↪255),2)# Add character description

    return img,mask,morphologyEx_img

with PiCamera() as camera:
    print("start color detect")
    camera.resolution = (640,480)
    camera.framerate = 24
    rawCapture = PiRGBArray(camera, size=camera.resolution)
    time.sleep(2)

    for frame in camera.capture_continuous(rawCapture, format="bgr",use_video_
↪port=True):# use_video_port=True
        img = frame.array
        img,img_2,img_3 =  color_detect(img,'red')  # Color detection function
        cv2.imshow("video", img)     # OpenCV image show
        cv2.imshow("mask", img_2)     # OpenCV image show
        cv2.imshow("morphologyEx_img", img_3)     # OpenCV image show
        rawCapture.truncate(0)    # Release cache

        k = cv2.waitKey(1) & 0xFF
        # 27 is the ESC key, which means that if you press the ESC key to exit
        if k == 27:
            break

    print('quit ...')
    cv2.destroyAllWindows()
    camera.close()
```

这个怎么运作？

首先将 HSV 颜色空间 中 H 的范围定义为字典，方便后面的颜色判断算法：

```
color_dict = {'red':[0,4],'orange':[5,18],'yellow':[22,37],'green':[42,85],'blue':[92,
↪110],'purple':[115,165],'red_2':[165,180]}
```

然后，定义大小为 5x5 的 卷积核 ，将用于形态学操作，如过滤。

```
kernel_5 = np.ones((5,5),np.uint8)
```

接下来，color_detect() 函数将分四步处理图片：

1. 提取目标颜色的数据作为新的二值图像（数组）。
2. 执行高级形态变换。
3. 在二值图像中寻找轮廓。
4. 在图像上为识别的对象绘制一个框架。

```
def color_detect(img,color_name):

    # The blue range will be different under different lighting conditions and can be␣
↪adjusted flexibly.  H: chroma, S: saturation v: lightness
    resize_img = cv2.resize(img, (160,120), interpolation=cv2.INTER_LINEAR)  # In␣
↪order to reduce the amount of calculation, the size of the picture is reduced to␣
↪(160,120)
    hsv = cv2.cvtColor(resize_img, cv2.COLOR_BGR2HSV)               # Convert from BGR␣
↪to HSV
    color_type = color_name

    mask = cv2.inRange(hsv,np.array([min(color_dict[color_type]), 60, 60]), np.
↪array([max(color_dict[color_type]), 255, 255]) )           # inRange(): Make the␣
↪ones between lower/upper white, and the rest black
    if color_type == 'red':
            mask_2 = cv2.inRange(hsv, (color_dict['red_2'][0],0,0), (color_dict['red_2
↪'][1],255,255))
            mask = cv2.bitwise_or(mask, mask_2)

    morphologyEx_img = cv2.morphologyEx(mask, cv2.MORPH_OPEN, kernel_5,iterations=1) ␣
↪            # Perform an open operation on the image

    # Find the contour in morphologyEx_img, and the contours are arranged according␣
↪to the area from small to large.
    _tuple = cv2.findContours(morphologyEx_img,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_
↪SIMPLE)
    # compatible with opencv3.x and openc4.x
    if len(_tuple) == 3:
        _, contours, hierarchy = _tuple
    else:
        contours, hierarchy = _tuple

    color_area_num = len(contours) # Count the number of contours

    if color_area_num > 0:
        for i in contours:    # Traverse all contours
            x,y,w,h = cv2.boundingRect(i)      # Decompose the contour into the␣
↪coordinates of the upper left corner and the width and height of the recognition␣
↪object

            # Draw a rectangle on the image (picture, upper left corner coordinate,␣
↪lower right corner coordinate, color, line width)
            if w >= 8 and h >= 8: # Because the picture is reduced to a quarter of␣
↪the original size, if you want to draw a rectangle on the original picture to␣
↪circle the target, you have to multiply x, y, w, h by 4.
                x = x * 4
                y = y * 4
```

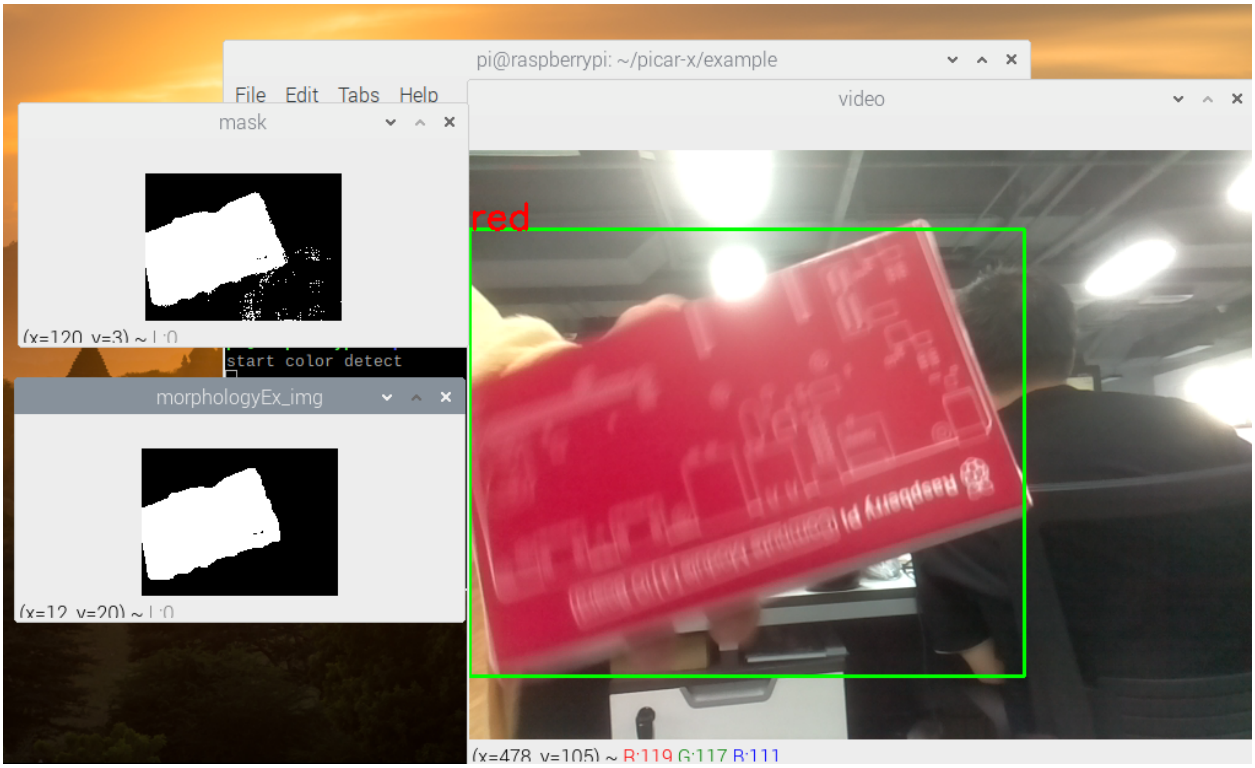
(续下页)

(接上页)

```
                w = w * 4
                h = h * 4
                cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),2)  # Draw a rectangular␣
↪frame
                cv2.putText(img,color_type,(x,y), cv2.FONT_HERSHEY_SIMPLEX, 1,(0,0,
↪255),2)# Add character description

    return img,mask,morphologyEx_img
```

img , mask , morphologyEx_img 显示在三个窗口中，可以直接观察每一步的处理结果。



5.9 人脸检测

这个项目也是基于计算机视觉 项目，增加了人脸检测算法。

运行代码

备注:

- 这个项目需要访问树莓派的桌面来查看相机模块拍摄的画面。
- 你可以将屏幕连接到 PiCar-X 上，或者参考教程远程桌面，用 VNC 或 XRDP 访问它。
- 一旦进入树莓派的桌面，打开 Terminal 并输入以下命令来运行它，或者直接用 Python 编辑器打开并运行它。

```
cd /home/pi/picar-x/example
sudo python3 human_face_detect.py
```

代码运行后，检测到的人脸将在屏幕中框选出。

代码

```
import cv2
from picamera.array import PiRGBArray
from picamera import PiCamera
import time


def human_face_detect(img):
    resize_img = cv2.resize(img, (320,240), interpolation=cv2.INTER_LINEAR)          #␣
↪In order to reduce the amount of calculation, resize the image to 320 x 240 size
    gray = cv2.cvtColor(resize_img, cv2.COLOR_BGR2GRAY)     # Convert to grayscale
    faces = face_cascade.detectMultiScale(gray, 1.3, 2)     # Detect faces on␣
↪grayscale images
    face_num = len(faces)    # Number of detected faces
    if face_num  > 0:
        for (x,y,w,h) in faces:

            x = x*2    # Because the image is reduced to one-half of the original size,
↪ the x, y, w, and h must be multiplied by 2.
            y = y*2
            w = w*2
            h = h*2
            cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)   # Draw a rectangle on the␣
↪face

    return img


with PiCamera() as camera:
    print("start human face detect")
    camera.resolution = (640,480)
    camera.framerate = 24
    rawCapture = PiRGBArray(camera, size=camera.resolution)
    time.sleep(2)

    for frame in camera.capture_continuous(rawCapture, format="bgr",use_video_
↪port=True): # use_video_port=True
        img = frame.array
        img =  human_face_detect(img)
        cv2.imshow("video", img)  #OpenCV image show
        rawCapture.truncate(0)  # Release cache

        k = cv2.waitKey(1) & 0xFF
        # 27 is the ESC key, which means that if you press the ESC key to exit
        if k == 27:
            break

    print('quit ...')
    cv2.destroyAllWindows()
    camera.close()
```

这个怎么运作？

在与此项目相同的路径 (picar-x/example/) 中，放置一个文件 haarcascade_frontalhumanface_default.xml。该文件是在 OpenCV 中训练的人脸检测模型文件。

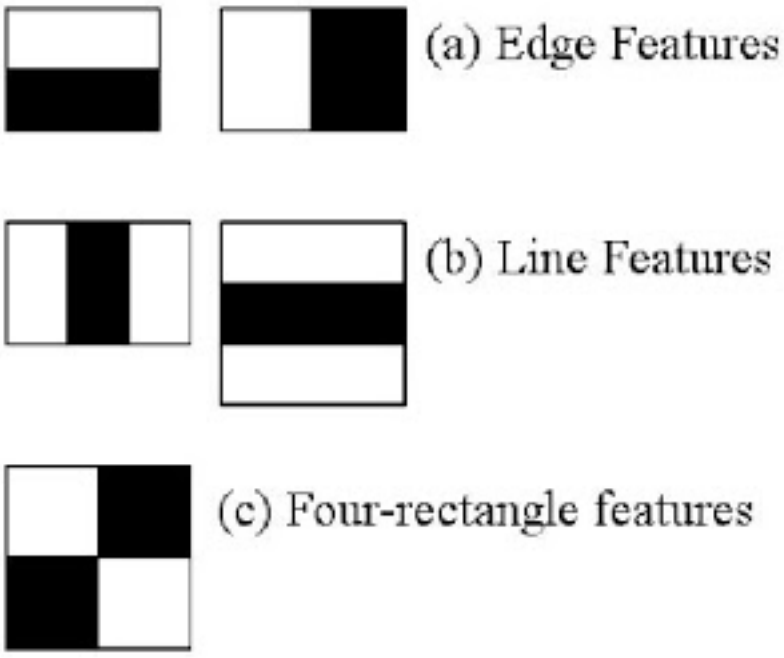
该文件由 OpenCV 的 **Cascade Classifier** 调用。

```
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
```

使用基于 Haar 特征的级联分类器的对象检测是 Paul Viola 和 Michael Jones 在他们的论文《Rapid Object Detection using a Boosted Cascade of Simple Features》于 2001 年提出的一种有效的对象检测方法。

这是一种基于机器学习的方法，从大量正负图像中训练级联函数，然后用于检测其他图像中的对象。

当使用人脸检测时，算法最初需要大量的正图像（人脸图像）和负图像（没有人脸的图像）来训练分类器。从那里，然后需要提取面部特征。为此，使用了下图中显示的 Haar 特征，类似于卷积核。每个特征是通过从黑色矩形下的像素总和中减去白色矩形下的像素总和获得的单个值。



- 级联分类器
- 级联分类器训练

human_human face_detect() 函数分三步处理图片：

1. 将图片转换为灰度。
2. 在灰度图像上检测人脸，得到检测人脸的边界矩形。
3. 在图像上为识别的对象绘制一个框架。

```
def human_face_detect(img):
    resize_img = cv2.resize(img, (320,240), interpolation=cv2.INTER_LINEAR)          #␣
↪In order to reduce the amount of calculation, resize the image to 320 x 240 size
    gray = cv2.cvtColor(resize_img, cv2.COLOR_BGR2GRAY)     # Convert to grayscale
    faces = face_cascade.detectMultiScale(gray, 1.3, 2)     # Detect faces on␣
↪grayscale images
    face_num = len(faces)    # Number of detected faces
```

(续下页)

(接上页)

```
    if face_num  > 0:
        for (x,y,w,h) in faces:

            x = x*2   # Because the image is reduced to one-half of the original size,
↪ the x, y, w, and h must be multiplied by 2.
            y = y*2
            w = w*2
            h = h*2
            cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)  # Draw a rectangle on the␣
↪face

    return img
```

- detectMultiScale - OpenCV

5.10 视频车

该程序将提供帕克的第一人称视角！使用键盘 WSAD 键控制移动方向，使用 O 和 P 调整速度。

运行代码

备注:

- 这个项目需要访问树莓派的桌面来查看相机模块拍摄的画面。
- 你可以将屏幕连接到 PiCar-X 上，或者参考教程远程桌面，用 VNC 或 XRDP 访问它。
- 一旦进入树莓派的桌面，打开 Terminal 并输入以下命令来运行它，或者直接用 Python 编辑器打开并运行它。

```
cd /home/pi/picar-x/example
sudo python3 video_car.py
```

代码运行后，您可以看到 PiCar-X 正在拍摄画面，并通过按以下键来控制它。

- O：加速
- P：减速
- W：前锋
- S：向后
- A：左转
- D：右转
- F：停止
- T：拍照
- ESC / Ctrl+C：退出

代码

```
# #!/usr/bin/env python3

print('Please run under desktop environment (eg: vnc) to display the image window')

from utils import reset_mcu
reset_mcu()
from picarx import Picarx
from vilib import Vilib
from time import sleep, time, strftime, localtime
import readchar

manual = '''
Press key to call the function(non-case sensitive):
    O: speed up
    P: speed down
    W: forward
    S: backward
    A: turn left
    D: turn right
    F: stop
    T: take photo
    ESC / Ctrl+C: quit
'''


px = Picarx()

def take_photo():
    _time = strftime('%Y-%m-%d-%H-%M-%S',localtime(time()))
    name = 'photo_%s'%_time
    path = "/home/pi/Pictures/picar-x/"
    Vilib.take_photo(name, path)
    print('\nphoto save as %s%s.jpg'%(path,name))


def move(operate:str, speed):

    if operate == 'stop':
        px.stop()
    else:
        if operate == 'forward':
            px.set_dir_servo_angle(0)
            px.forward(speed)
        elif operate == 'backward':
            px.set_dir_servo_angle(0)
            px.backward(speed)
        elif operate == 'turn left':
            px.set_dir_servo_angle(-30)
            px.forward(speed)
        elif operate == 'turn right':
            px.set_dir_servo_angle(30)
            px.forward(speed)



def main():
    speed = 0
```

(续下页)

(接上页)

```
    status = 'stop'

    Vilib.camera_start(vflip=False,hflip=False)
    Vilib.display(local=True,web=True)
    sleep(2)  # wait for startup
    print(manual)

    while True:
        print("\rstatus: %s , speed: %s    "%(status, speed), end='', flush=True)
        # readkey
        key = readchar.readkey().lower()
        # operation
        if key in ('wsadfop'):
            # throttle
            if key == 'o':
                if speed <=90:
                    speed += 10
            elif key == 'p':
                if speed >=10:
                    speed -= 10
                if speed == 0:
                    status = 'stop'
            # direction
            elif key in ('wsad'):
                if speed == 0:
                    speed = 10
                if key == 'w':
                    # Speed limit when reversing,avoid instantaneous current too large
                    if status != 'forward' and speed > 60:
                        speed = 60
                    status = 'forward'
                elif key == 'a':
                    status = 'turn left'
                elif key == 's':
                    if status != 'backward' and speed > 60: # Speed limit when␣
↪reversing
                        speed = 60
                    status = 'backward'
                elif key == 'd':
                    status = 'turn right'
            # stop
            elif key == 'f':
                status = 'stop'
            # move
            move(status, speed)
        # take photo
        elif key == 't':
            take_photo()
        # quit
        elif key == readchar.key.CTRL_C or key in readchar.key.ESCAPE_SEQUENCES:
            print('\nquit ...')
            px.stop()
            Vilib.camera_close()
            break

        sleep(0.1)
```

(续下页)

(接上页)

```
if __name__ == "__main__":
    try:
        main()
    except Exception as e:
        print("error:%s"%e)
    finally:
        px.stop()
        Vilib.camera_close()
```

CHAPTER 6

附录

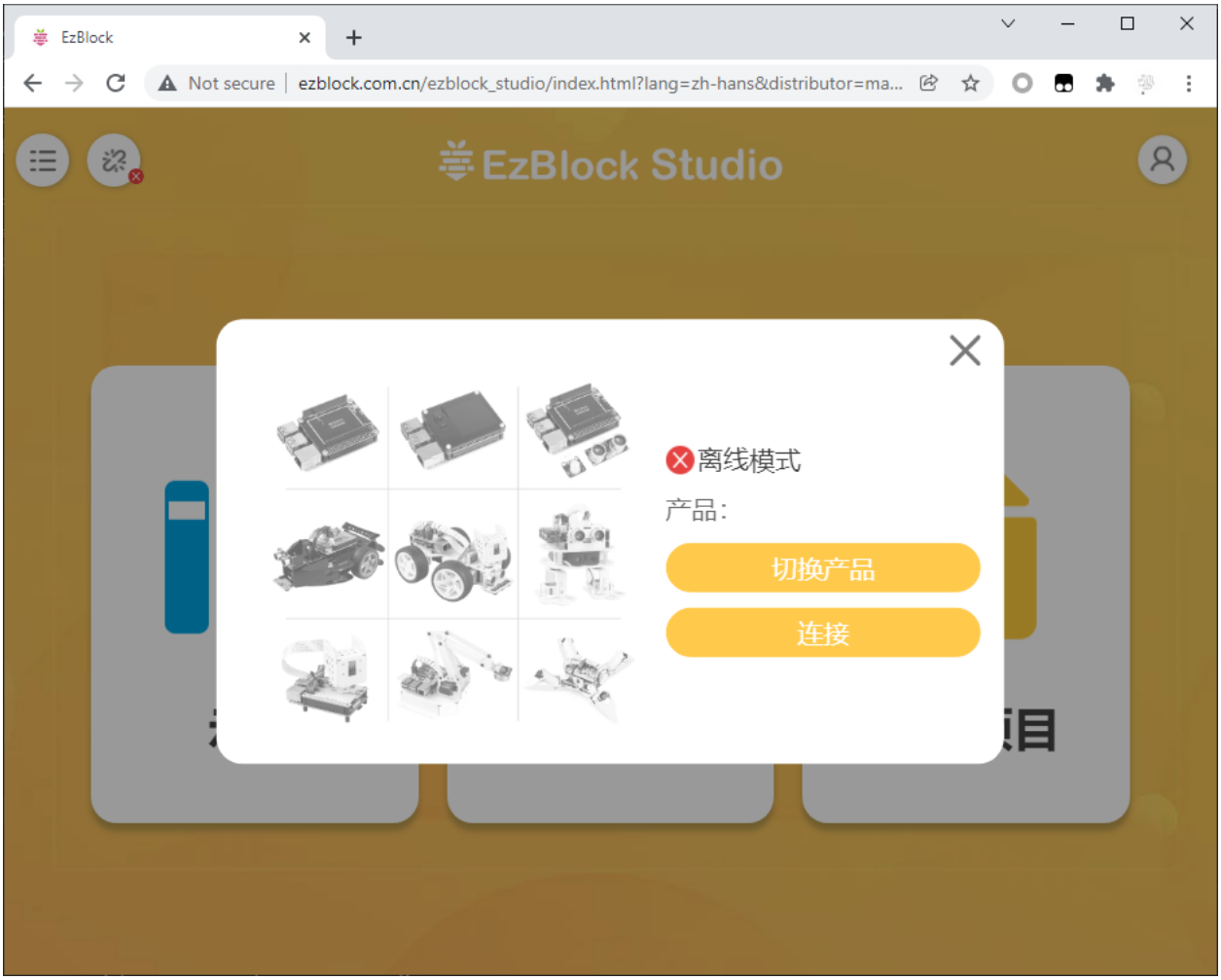
6.1 网页版 EzBlock

进入 网页版 EzBlock 。

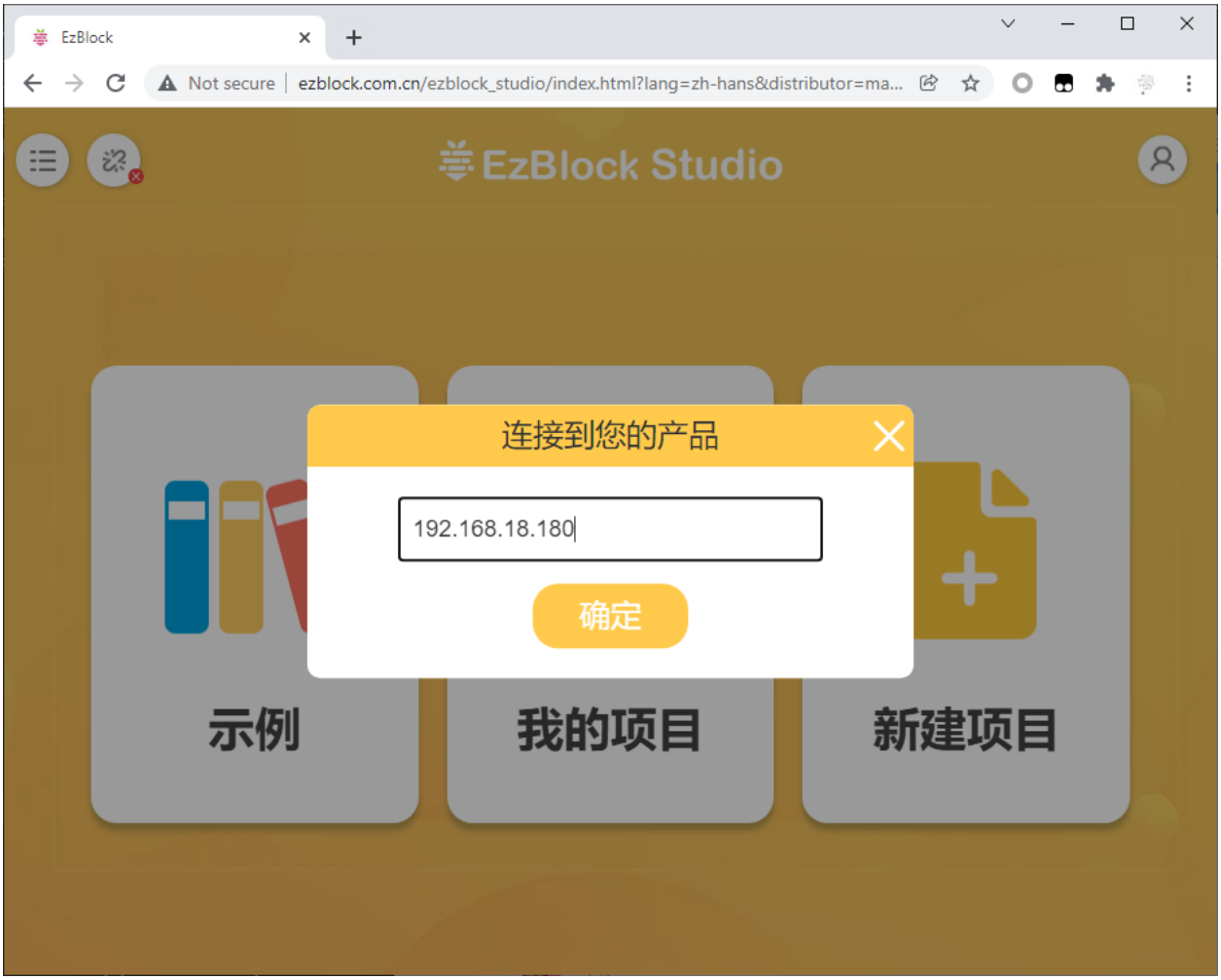
在首页点击左上角的产品按钮。

EzBlock
Not secure
ezblock.com.cn/ezblock_studio/index.html?lang=zh-hans&distributor=ma...
EzBlock Studio
示例
我的项目
新建项目

点击连接。


EzBlock
Not secure
ezblock.com.cn/ezblock_studio/index.html?lang=zh-hans&distributor=ma...
EzBlock Studio
离线模式
产品:
切换产品
连接

输入 IP 地址。



随后将弹出“连接成功”的提示，此时你可以回到*玩转 EzBlock* 。

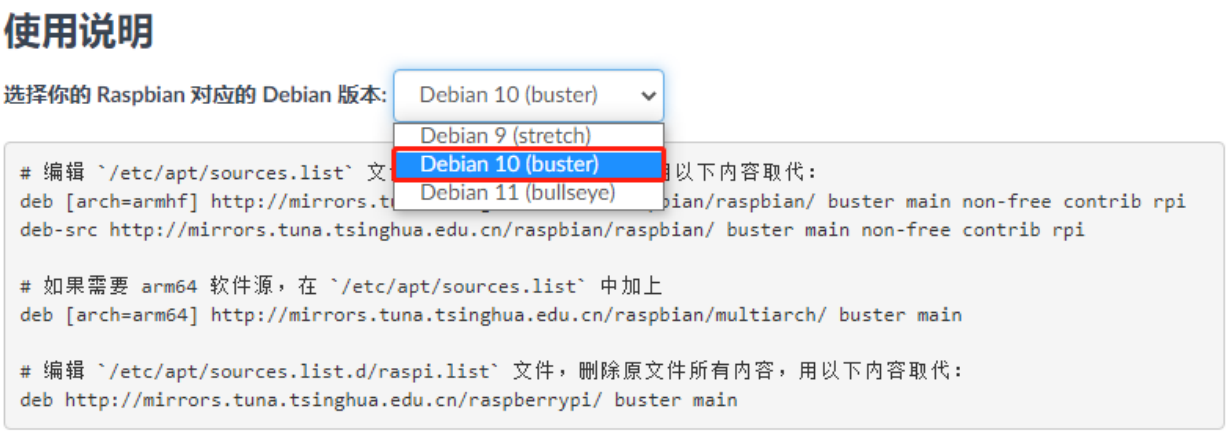
6.2 apt 和 pip 更换国内源

树莓派系统默认的 apt 源和 pip 源都是国外的服务器，使用国内网络访问可能会发生超时（ReadTimeoutErro），或被拒绝访问的情况，如果是这样我们可以将 apt 和 pip 更改为国内的源，步骤如下所示：

1.apt 更换国内源

1）访问链接：https://mirrors.tuna.tsinghua.edu.cn/help/raspbian/ 以了解配置文件修改详情。

2）不同的树莓派系统版本修改不同，先选择对应的版本，比如我的是 Debian 10 （buster），如果你的树莓派系统是 bullseye 则选择对应版本，若没有符合的版本则请重新安装 buster 及以下版本的系统。



备注: 一般不删除原内容，可以将其用 # 注释掉。

3）我们这里以 raspbian buster 版本为例，用 nano 命令打开 `/etc/apt/sources.list` 文件。

```
sudo nano /etc/apt/sources.list
```

4）然后用 # 将原本的内容注释掉，在最后面附上下面的代码。

```
# deb http://raspbian.raspberrypi.org/raspbian/ buster main contrib non-free rpi
# Uncomment line below then 'apt-get update' to enable 'apt-get source'
#deb-src http://raspbian.raspberrypi.org/raspbian/ buster main contrib non-free rpi

deb [arch=armhf] http://mirrors.tuna.tsinghua.edu.cn/raspbian/raspbian/ buster main␣
↪non-free contrib rpi
deb-src http://mirrors.tuna.tsinghua.edu.cn/raspbian/raspbian/ buster main non-free␣
↪contrib rpi
```

5）按下 `Ctrl+O` 保存，按下 `Ctrl+X` 和 `Y` 退出。

6）用 nano 命令打开 `etc/apt/sources.list.d/raspi.list` 文件。

```
sudo nano /etc/apt/sources.list.d/raspi.list
```

7）然后用 # 将原本的内容注释掉，在最后面附上 deb... 代码。

```
# deb http://archive.raspberrypi.org/debian/ buster main
# Uncomment line below then 'apt-get update' to enable 'apt-get source'
#deb-src http://archive.raspberrypi.org/debian/ buster main

deb http://mirrors.tuna.tsinghua.edu.cn/raspberrypi/ buster main ui
```

8）按下 `Ctrl+O` 保存，按下 `Ctrl+X` 和 `Y` 退出。

9）用以下命令更新软件列表:

```
sudo apt update
```

2.(Pypi) pip 更换国内源

可以参考链接 https://mirrors.tuna.tsinghua.edu.cn/help/pypi/ 了解详情。

有两种方法修改配置文件。

方法一：使用 pip 指令设置

```
pip3 config set global.index-url https://pypi.tuna.tsinghua.edu.cn/simple
```

方法二：手动编辑文件

Linux/Mac os 环境中，配置文件位置在 ~/.pip/pip.conf（如果不存在则手动创建该目录和文件）。

```
sudo mkdir -p ~/.pip
sudo nano ~/.pip/pip.conf
```

然后按如下编辑文件内容

```
[global]
index-url = https://pypi.tuna.tsinghua.edu.cn/simple
[install]
trusted-host = https://pypi.tuna.tsinghua.edu.cn
```

最后 apt 和 pip 都已经更换了国内下载源，这样下载速度就会提高很快不会导致下载失败了。

6.3 Filezilla 软件

文件传输协议 (FTP) 是一种标准通信协议，用于在计算机网络上将计算机文件从服务器传输到客户端。

Filezilla 是一款开源软件，不仅支持 FTP，还支持 FTP over TLS (FTPS) 和 SFTP。我们可以使用 Filezilla 将本地文件（如图片和音频等）上传到树莓派，或者从树莓派下载文件到本地。

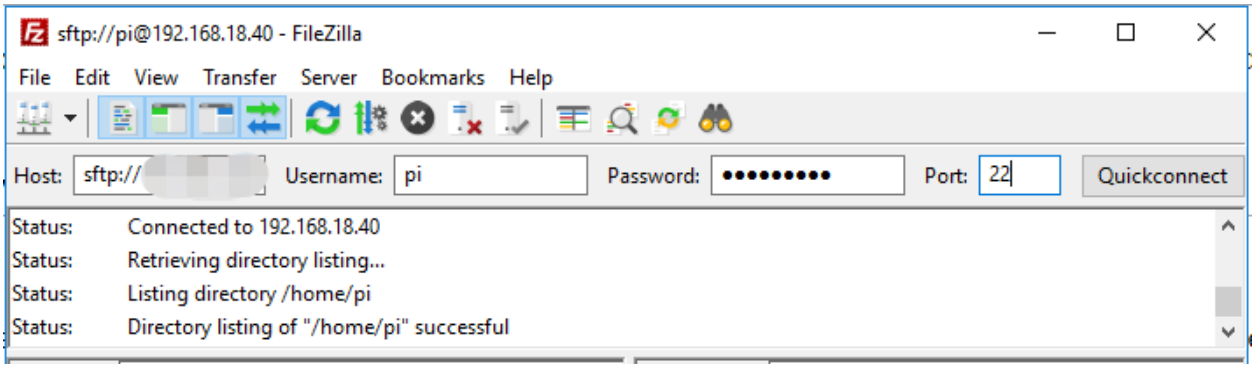
第 1 步: 下载 Filezilla。

在 Filezilla 官方网站 下载客户端, Filezilla 有一个很好的教程，请参考: Filezilla 文档 。

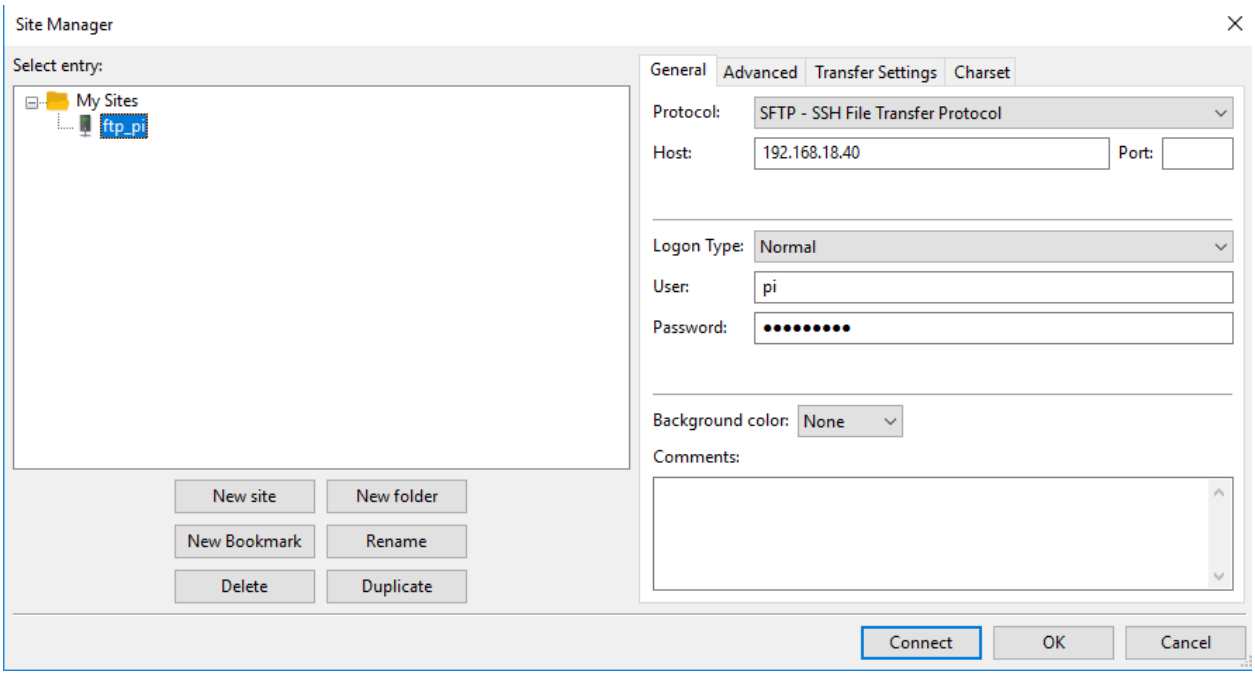
第 2 步: 连接树莓派

快速安装后打开它，现在 连接到 FTP 服务器。

它有 3 种连接方式，这里我们使用 **快速连接**栏。输入 **主机名/IP** 、**用户名**、**密码**和 **端口（22）**，然后点击 **快速连接**或按 **回车**连接到服务器。

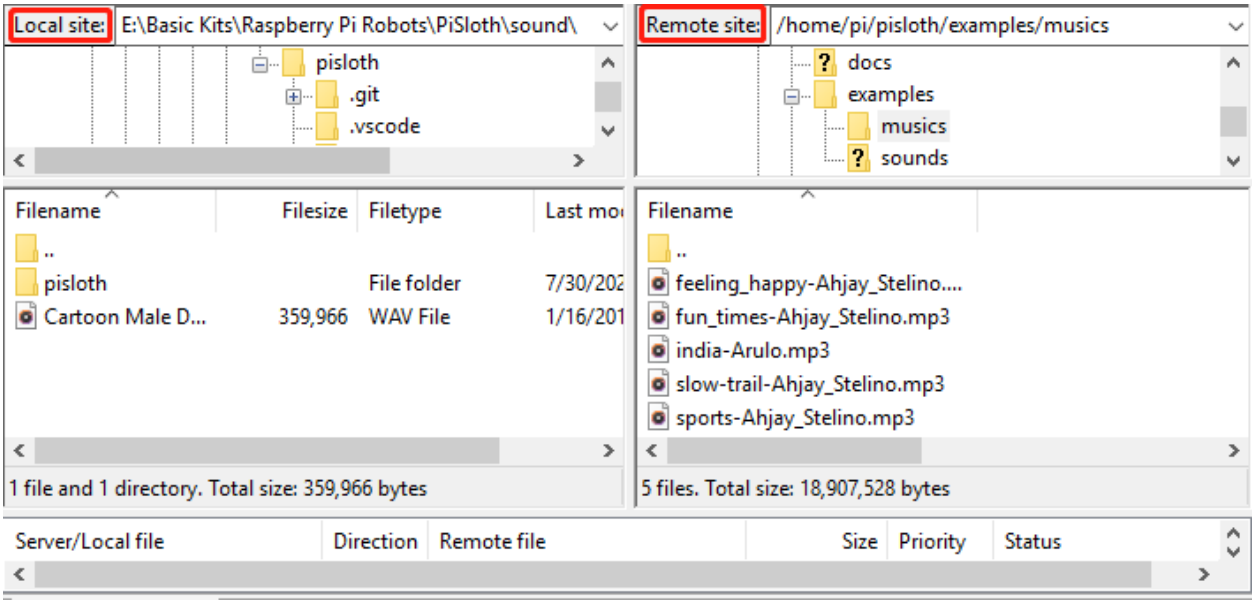


备注: 快速连接是测试您的登录信息的好方法。如果要创建永久条目，可以在快速连接成功后选择 **File-> Copy Current Connection to Site Manager** （文件-> 将当前连接复制到站点管理器），输入名称并单击 **确定**。下次您将能够通过在 **File -> Site Manager** (文件-> 站点管理器。) 中选择先前保存的站点进行连接。



第 3 步: 上传/下载文件。

您可以通过拖放将本地文件上传到树莓派，也可以将树莓派文件中的文件下载到本地。



6.4 关于电池

适用参数

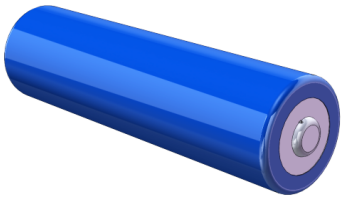
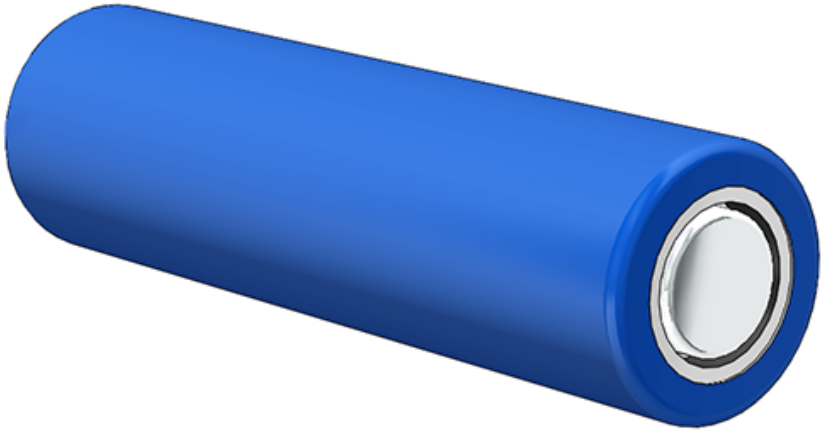
- 3.7V
- 18650
- 可充电的
- 锂离子电池
- 尖头电池
- 无保护板

备注:

- Robot HAT 无法给电池充电，需要购买电池充电器。
- 当 Robot HAT 上的两个电量指示灯熄灭时，表示电量过低，需要给电池充电。

尖头 vs 平头？

请选择尖头电池，以确保电池与电池座之间的连接良好。

尖头	平头
	

没有保护板？

建议使用没有保护板的 18650 电池。否则可能会因为保护板的过流保护而导致机器人断电停止运行。

电池容量？

为了让机器人长时间工作，尽量使用大容量电池。建议购买容量为 3000mAh 及以上的电池。

6.5 PuTTY

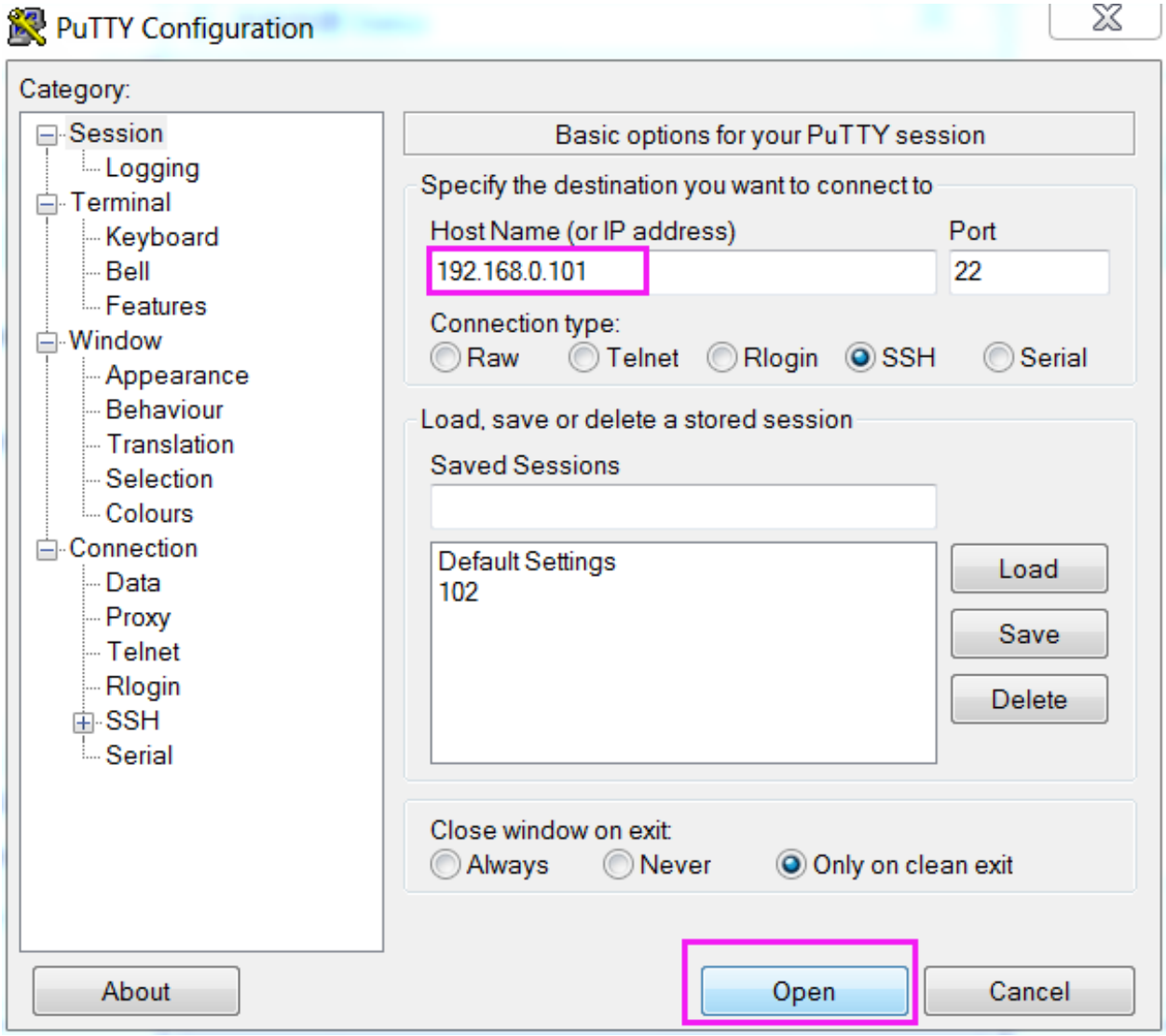
如果你是 Windows 用户，你可以使用一些 SSH 应用程序。在这里，我们推荐 PuTTY。

步骤 1

下载 PuTTY。

步骤 2

打开 PuTTY，点击左侧的树状结构中的 **会话 (Session)**。在 **主机名 (或 IP 地址)（Host Name (or IP address)）** 下的文本框中输入 RPi 的 IP 地址，并在 **端口 (Port)** 下输入 **22** （默认为 22）。



步骤 3

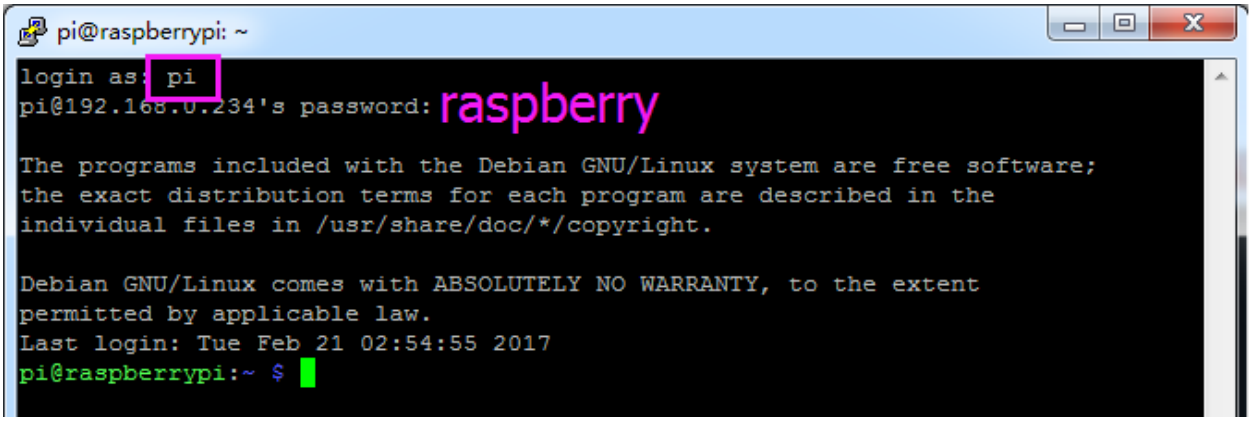
点击 **打开 (Open)**。请注意，当你首次使用 IP 地址登录树莓派时，会提示一个安全提醒。只需点击 **是 (Yes)**。

步骤 4

当 PuTTY 窗口提示 **登录为 (login as):**，输入 **pi** （RPi 的用户名）和 **密码 (password)**。

备注： 当你输入密码时，字符不会在窗口上显示，这是正常的。你需要做的是输入正确的密码。

如果 PuTTY 旁边显示为不活动 (inactive)，意味着连接已经断开，需要重新连接。



步骤 5

在这里，我们已经连接上了树莓派，现在是进行下一步的时候了。

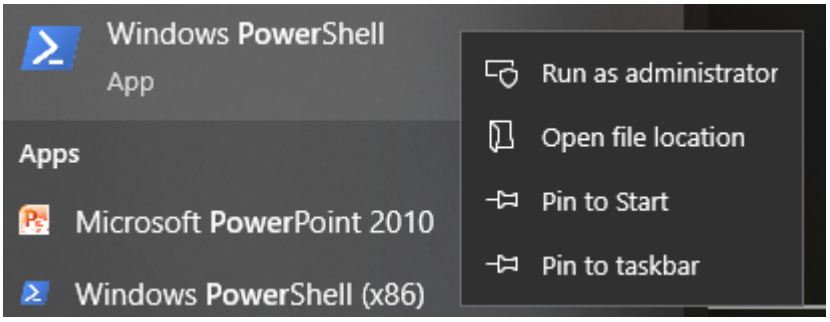
6.6 通过 Powershell 安装 OpenSSH

当你使用 `ssh < 用户名>@< 主机名>.local`` (或 ``ssh < 用户名>@<IP 地址>`) 连接到你的树莓派时，但以下错误信息出现。

```
ssh: "ssh
↪"不被识别为cmdlet、函数、脚本文件或可运行程序的名称。检查名称的拼写，或者如果包含了路径，请确认
```

这意味着你的计算机系统太旧，没有预先安装 'OpenSSH <https://learn.microsoft.com/en-us/windows-server/administration/openssh/openssh_install_firstuse?tabs=gui>'_，你需要按照下面的教程手动安装它。

1. 在你的 Windows 桌面的搜索框中输入 `powershell`，右击 `Windows PowerShell`，然后从出现的菜单中选择 以管理员身份运行。



2. 使用以下命令安装 `OpenSSH.Client`。

```
Add-WindowsCapability -Online -Name OpenSSH.Client~~~~0.0.1.0
```

3. 安装后，将返回以下输出。

```
Path          :
Online        : True
RestartNeeded : False
```

4. 使用以下命令验证安装。

```
Get-WindowsCapability -Online | Where-Object Name -like 'OpenSSH*'
```

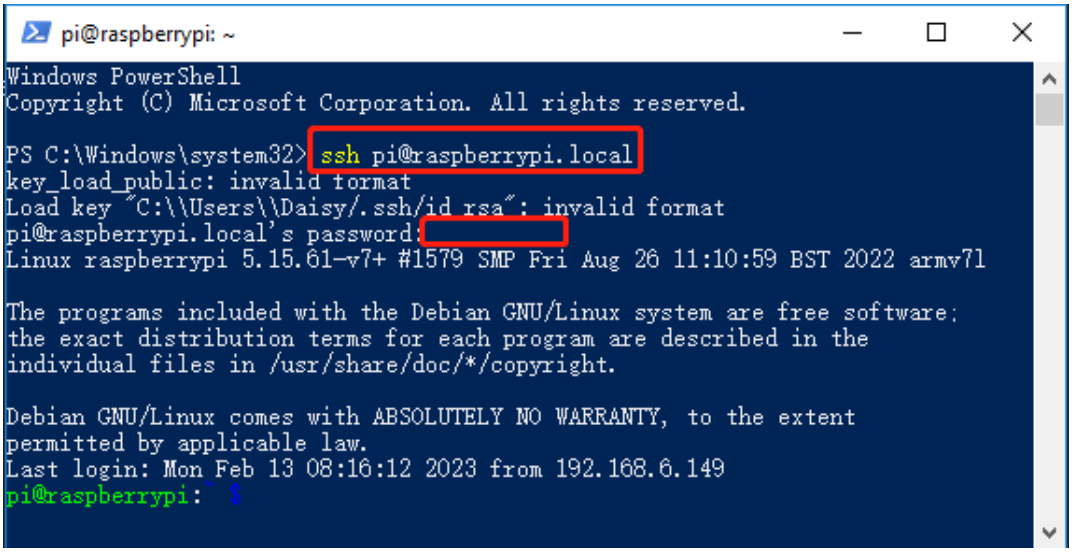
5. 现在它告诉你 `OpenSSH.Client` 已成功安装。

```
Name  : OpenSSH.Client~~~~0.0.1.0
State : Installed

Name  : OpenSSH.Server~~~~0.0.1.0
State : NotPresent
```

> **警告：** 如果上面的提示没有出现，意味着你的 Windows 系统仍然太旧，建议你安装第三方 SSH 工具，如*PuTTY*。

6. 现在重启 PowerShell 并继续以管理员身份运行它。此时，你将能够使用 `ssh` 命令登录到你的树莓派，在那里你将被提示输入你之前设置的密码。



CHAPTER 7

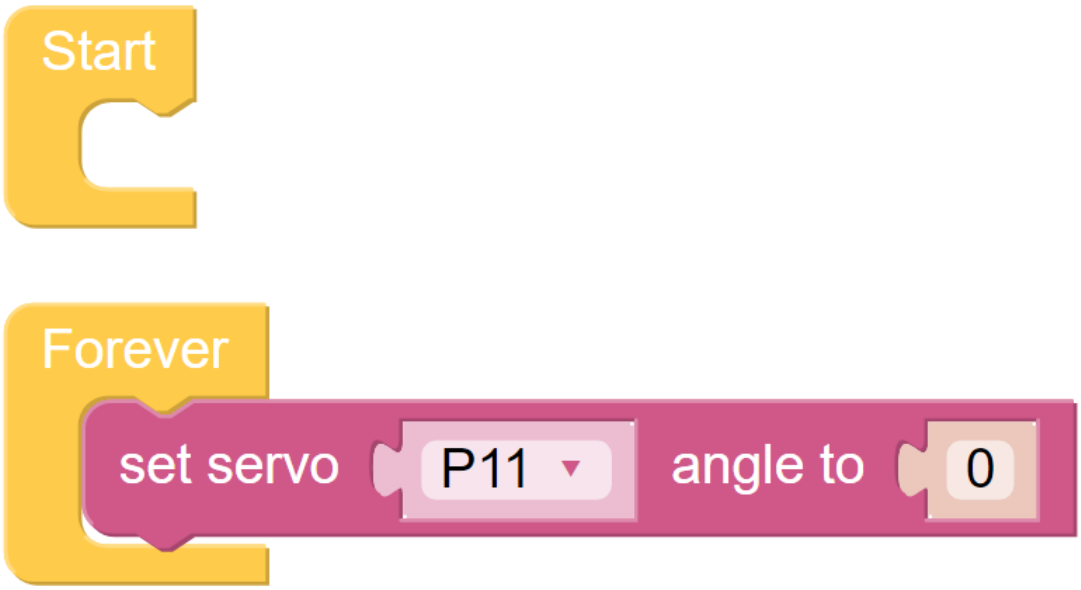
疑难解答

7.1 当使用 VNC 时，我被提示桌面暂时无法显示？

在终端，输入 `sudo raspi-config` 来改变分辨率。

7.2 安装 EzBlock 操作系统后，舵机不能转到 0°？

- 检查伺服电缆是否正确连接，Robot HAT 的电源是否打开。
- 按下复位按钮。
- 如果您已经在 EzBlock Studio 中运行了程序，那么 P11 的自定义程序就不再可用。您可以参考下图，在 EzBlock Studio 中手动编写程序，将伺服角度设置为 0。



7.3 如何在 EzBlock 重新校准机器人

在首页，点击左上角的产品连接图标。

进入到产品页面之后，点击 **设置**按钮。

进入设置页面后，点击 **校准**按钮就能进入到校准页面，然后按照提示来校准。

7.4 EzBlock 无法连接蓝牙？

- 先检查是否有给你的 SD 卡烧录*安装 EzBlock 镜像*。
- 打开产品的电源开关后，等蓝牙指示灯变得更亮并且出现”zi~” 声后，代表树莓派成功启动，此时再去连接。
- 检查您的移动设备的蓝牙是否打开。
- 检查 EzBlock 是否被允许访问设备的位置。
- 有些移动设备还需要打开位置服务。
- 检查电池电量。如果两个电源指示灯都关闭，或者只有一个指示灯在闪烁；电源电量低，请给电池充电。

- 如果以上方法都试过了，请尝试按下 RST 按钮，或重新启动产品和 APP。

7.5 APP 搜索到蓝牙，但无法连接

- 先检查是否有给你的 SD 卡烧录*安装 EzBlock 镜像*。
- 打开产品的电源开关后，等蓝牙指示灯变得更亮并且出现”zi~” 声后，代表树莓派成功启动，此时再去连接。
- 检查 ROBOT HAT 上的 BLE 或 USR 灯是否一直亮着（这意味着产品被其他设备连接），如果是，请断开其他设备的连接或重新启动产品。
- 如果以上方法都试过了，请尝试按下 RST 按钮，或者重新启动产品和 APP。

7.6 配置 WIFI 后 APP 无法连接

- 检查国家、账号和密码是否正确。
- 检查该 WIFI 的网络状态。
- 检查电源电量。如果两个电源指示灯都熄灭或只有一个电源指示灯闪烁，则说明电源电量不足，请给电池充电。
- 检查配置的 WiFi 和移动设备连接的 WiFi 是否相同。

7.7 为什么伺服机有时会无缘无故地返回到中间位置？

当舵机被结构物或其他物体挡住，无法到达预定位置时，舵机会进入断电保护模式，以防止舵机被过大的电流烧坏。

断电一段时间后，如果没有给伺服机提供 PWM 信号，伺服机将自动恢复到原来的位置。

CHAPTER 8

版权声明

本手册中包括但不限于文字、图片、代码等所有内容均归 SunFounder 公司所有。根据相关规定和版权法，您只能将其用于个人学习、调查、欣赏或其他非商业或非营利目的，不得侵犯作者和相关权利人的合法权利。对于任何个人或组织未经许可将其用于商业利益，本公司保留采取法律行动的权利。