

产品简介

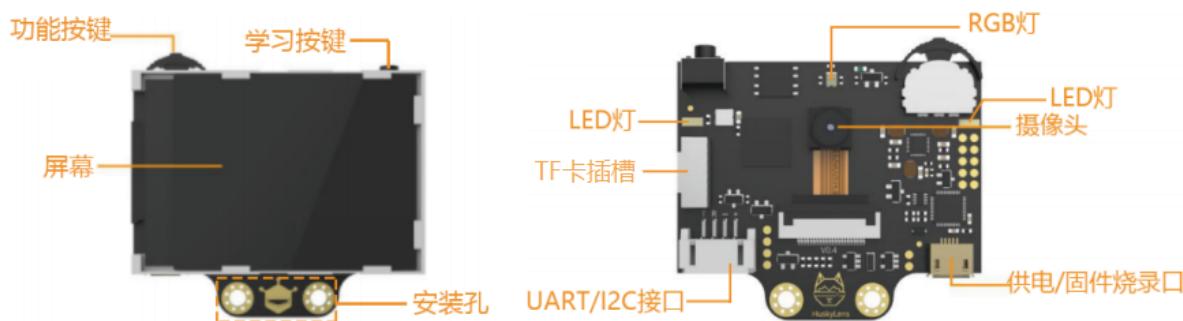
HuskyLens二哈识图是一款简单易用的AI视觉传感器，内置6种功能：人脸识别、物体追踪、物体识别、巡线追踪、颜色识别、标签(二维码)识别。仅需一个按键即可完成AI训练，摆脱繁琐的训练和复杂的视觉算法，让你更加专注于项目的构思和实现。

HuskyLens板载UART / I2C接口，可以连接到Arduino、Raspberry Pi、LattePanda、micro:bit等主流控制器，实现硬件无缝对接。HuskyLens直接输出识别结果给控制器，你无需折腾复杂的算法，就能制作非常有创意的项目。

技术规格

- 处理器：Kendryte K210
- 图像传感器：
 - SEN0305 HuskyLens: OV2640(200W像素)
 - SEN0336 HuskyLens PRO: OV5640(500W像素)
- 供电电压：3.3~5.0V
- 电流消耗：320mA@3.3V, 230mA@5.0V (电流值为典型值；人脸识别模式；80%背光亮度；补光灯关闭)
- 连线接口：UART, I2C
- 显示屏：2.0寸IPS，分辨率320*240
- 内置功能：人脸识别，物体追踪，物体识别，巡线追踪，颜色识别，标签识别
- 尺寸：52mm*44.5mm

接口与引脚说明



- 功能按键：左右拨动“功能按键”，切换功能
- 学习按键：将HuskyLens对准指定物体，按下“学习按键”即可学习该物体
- USB接口：连接到电源，给HuskyLens供电；也可以连接到电脑的USB接口，给HuskyLens升级固件

- 4pin接口 (UART模式)

| 序号 | 标注 | 管脚功能 | 用途 |
|----|----|------|--------------------|
| 1 | T | TX | HuskyLens的串口数据发送管脚 |
| 2 | R | RX | HuskyLens的串口数据接收管脚 |
| 3 | - | GND | 电源负极 |
| 4 | + | VCC | 电源正极 |

- 4pin接口 (I2C模式)

| 序号 | 标注 | 管脚功能 | 用途 |
|----|----|------|-------|
| 1 | T | SDA | 串行数据线 |
| 2 | R | SCL | 串行时钟线 |
| 3 | - | GND | 电源负极 |
| 4 | + | VCC | 电源正极 |

小提示:

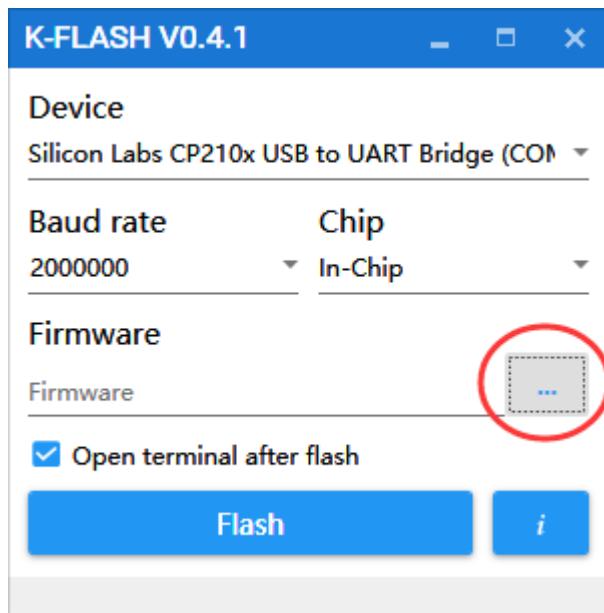
- USB接口与4pin接口，任接一个，即可对HuskyLens供电。
- 由于板载了电源自动切换电路，USB接口与4pin接口可以同时接电源，且优先使用USB接口上的电源。
- 请确保电源电压与功率足够，防止HuskyLens工作异常。

升级固件

使用本产品之前，强烈建议你升级到最新版本的固件，特别是Kickstarter的众筹支持者。

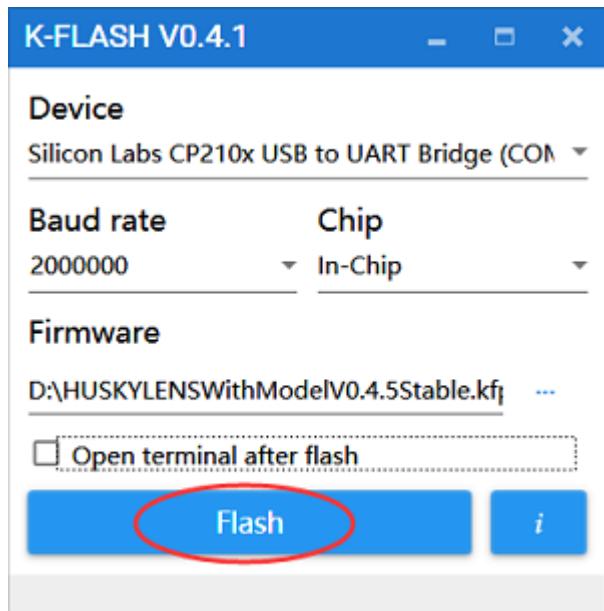
升级固件的步骤如下：

1. 下载K-Flash软件。 [点击此处](#) 下载。
2. 下载USB转串口芯片CP2102的驱动程序。 [点击此处](#) 选择适合您的驱动。
3. 下载最新的固件。 [点击此处](#) 查看所有版本的固件。 在本教程中，使用 **HUSKYLENSWithModelV0.4.5Stable.kfpkg** 固件。
4. 运行K-FLASH软件，然后点击“...”按钮，加载固件。

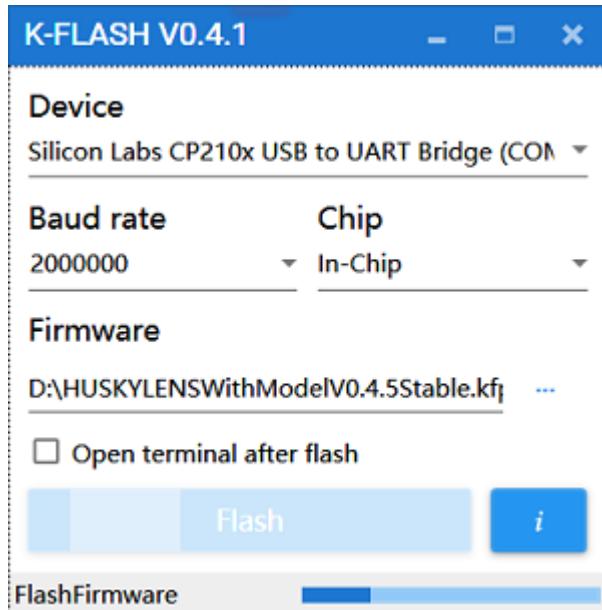


5. 请按照以下参数值，来设置 K-Flash软件:

- Device: 选择您电脑上的COM口，其名字含有 “Silicon Labs CP210x USB to UART Bridge” 字样
- Baud rage: 2000000
- Chip: In-Chip
- 不勾选 "Open terminal after flash"



6. 点击 “Flash” 按钮。烧录过程大概需要5分钟左右。固件中有大更新，因此上传时间会有点长。

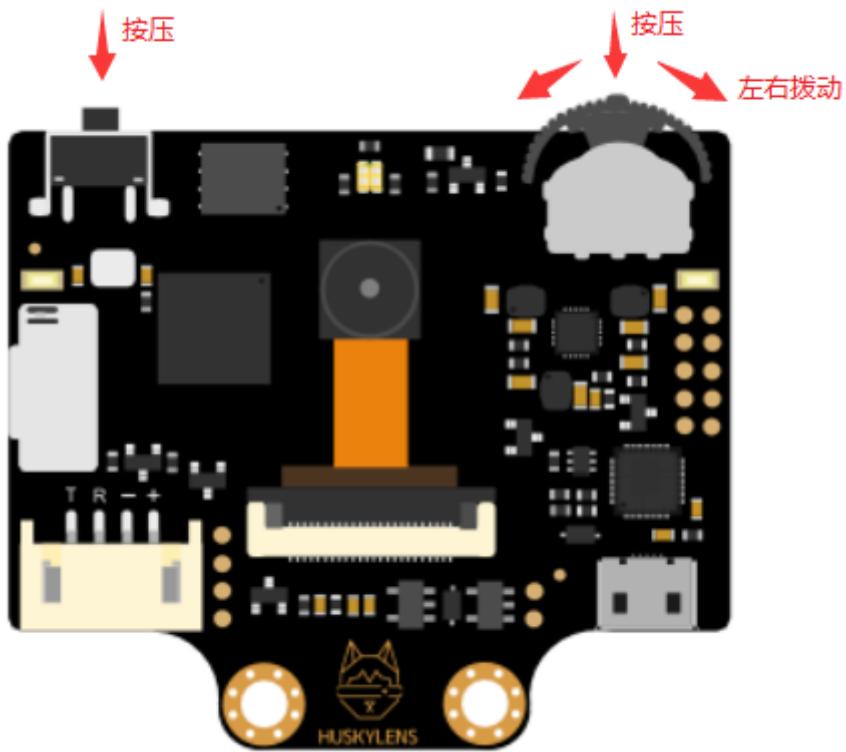


7. 出现弹窗，提示“Successful”，则固件的升级就完成了。

功能按键与学习按键

功能按键与学习按键的基本操作如下：

- 向左或向右拨动“功能按键”，切换到不同的功能
- 短按“学习按键”，学习指定的物体；长按“学习按键”，从不同的角度和距离持续学习指定的物体；如果HuskyLens之前学习过，则短按“学习按键”，可让HuskyLens忘记当前功能下所学的
- 长按“功能按键”，进入当前功能的二级菜单参数设置界面，向左、向右拨动或向下短按“功能按键”即可设置相关参数



设置中文语言

菜单界面的默认语言为英文。本教程使用中文语言界面。要将菜单界面的语言设置为中文，请按照如下步骤操作：

1. 向右拨动“功能按键”，至屏幕顶端显示“General Settings”。
2. 向下短按或长按“功能按键”，进入“General Settings”的二级菜单参数设置界面。
3. 向右拨动“功能按键”，选择“Language”参数（位于最后一个），然后短按“功能按键”，再向右拨动“功能按键”选择“简体中文”，再短按“功能按键”，此时HuskyLens会自动切换到中文语言。

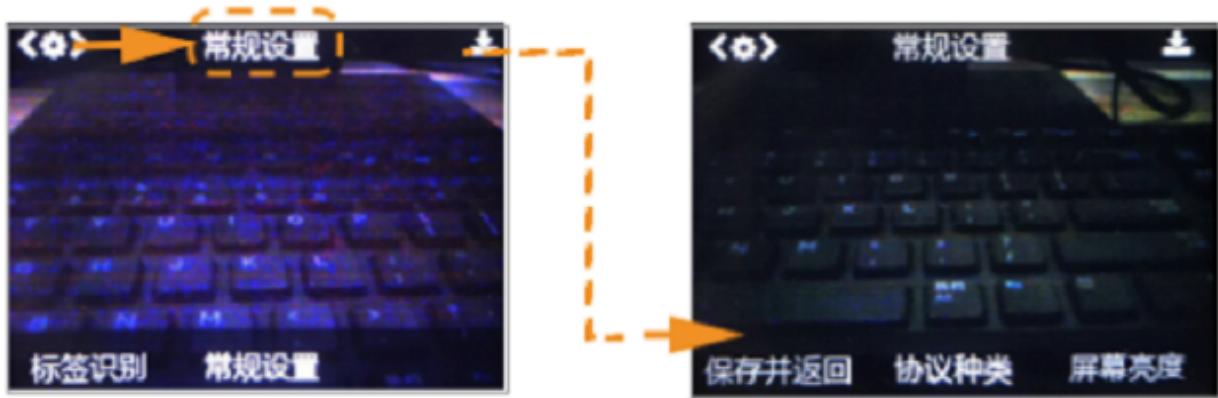


常规设置

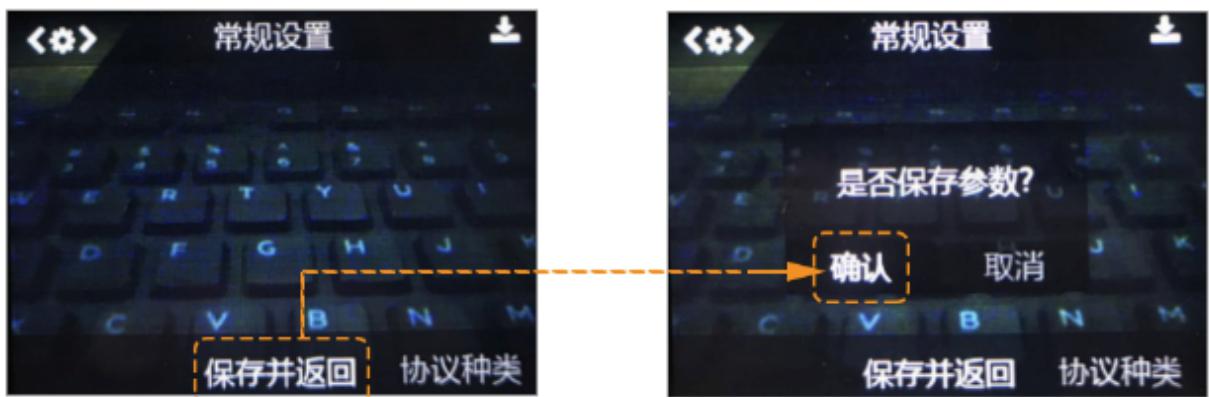
进入常规设置的二级菜单并调整各个参数的操作方法如下：

1. 选择“常规设置”：向右拨动“功能按键”，至屏幕顶端显示“常规设置”。

2. 进入“常规设置”界面：向下短按或长按“功能按键”，即可进入



3. 调整各个参数：向左或向右拨动“功能按键”，选择需要设置的参数，然后短按“功能按键”进入所选参数的设置，向左或向右拨动“功能按键”调整所选参数，再短按“功能按键”，确认该参数
4. 保存并退出：参数调整完毕后，向左拨动“功能按键”，直至选择“保存并返回”，然后短按“功能按键”，屏幕上会显示“是否保存参数”的提示，默认选择“确认”，此时，短按“功能按键”即可保存设置好的参数，并自动退出“常规设置”的二级菜单界面。



小提示：

- 常规设置里的各项参数，都是通过左右拨动“功能按键”和按下“功能按键”来进行选择和设置的。
- 设置好参数之后，务必选择“保存并返回”，才能将设置好的参数保存下来

常规设置的二级菜单参数设置界面，包含了10个不同的参数。

- **协议种类**

HuskyLens支持三种不同波特率的串口协议（9600,115200,1000000），也支持I2C协议。此外，还支持自动识别协议，也就是，HuskyLens会自动切换成串口或I2C协议。建议使用自动识别协议，简单方便。默认设置为自动识别。

- **屏幕亮度**

屏幕可以调节亮度，范围为1~100。默认值为80。

- **菜单自动隐藏**

如果有一段时间不操作HuskyLens，那么屏幕上的菜单会自动隐藏。你可以设置这个等待时间，范围为1~100秒。默认值为10秒。

- **LED开关**

HuskyLens板载了2个LED补光灯。你可以打开或者关闭补光灯。默认值为关。

- **LED亮度**

LED补光灯可以调节亮度，范围为1~100。默认值为50。

- **RGB开关**

HuskyLens板载了一颗RGB指示灯，你可以打开或者关闭RGB指示灯。默认值为开。

- **RGB亮度**

RGB指示灯也可以调节亮度，范围为1~100。默认值为20。

- **初始化设置**

通过初始化设置，把HuskyLens恢复到出厂状态

- **版本号**

当前固件的版本信息

- **语言**

目前HuskyLens支持中文和英文两种语言。通过语言设置，选择你需要的语言。

颜色指示定义

各个功能中，屏幕中央的方框、“+”字的颜色定义是统一的，有助于你了解当前HuskyLens所处的状态。

| 颜色 | 状态 |
|--------------------|-----------------|
| 由橙色渐变到黄色，再由黄色渐变到橙色 | 未学习过，待学习 |
| 黄色 | 正在学习中 |
| 蓝色 | 已学习过，且识别到学习过的东西 |

RGB指示灯目前仅用于人脸识别功能的状态指示，颜色定义如下：

| 颜色 | 状态 |
|----|----|
|----|----|

| | |
|----|-----------------|
| 蓝色 | 未学习过，侦测到人脸 |
| 黄色 | 正在学习中 |
| 绿色 | 已学习过，且识别到学习过的人脸 |

人脸识别

本功能可以侦测任何脸部轮廓；识别、追踪学习过的人脸。

识别单个人脸

默认设置为学习并识别单个人脸。

操作设置

向左拨动“功能按键”，直至屏幕顶部显示“人脸识别”。

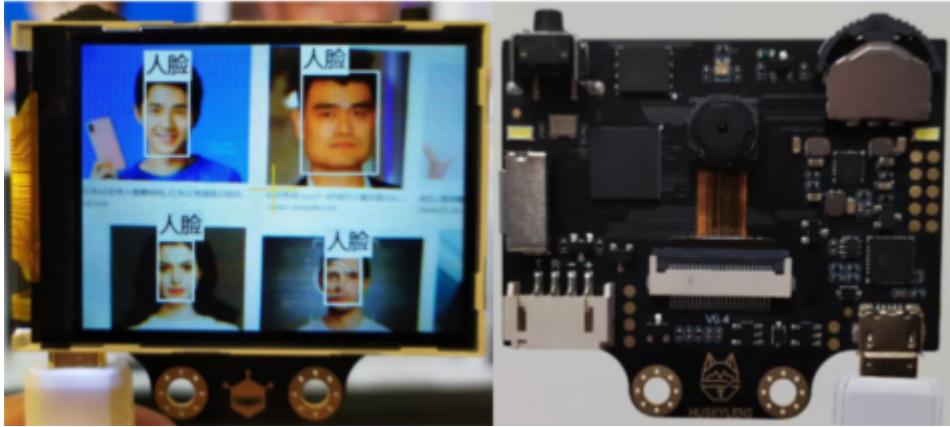


学习与识别

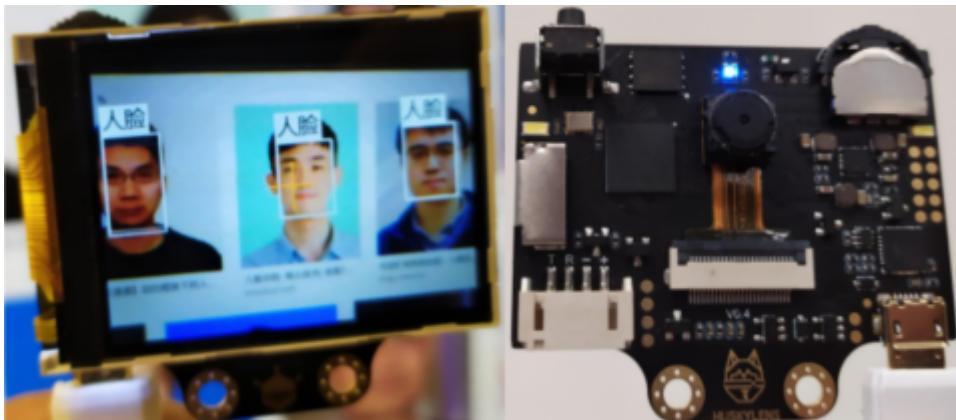
1. 侦测人脸：

把HuskyLens对准有人脸的区域，屏幕上会用白色框自动框选出检测到的所有人脸，并分别显示“人脸”字样。

若此时屏幕中央的“+”字没有对准任何人脸框，则另一面的RGB灯不亮。



若此时屏幕中央的“+”字对准在任意人脸框内，则另一面的RGB灯亮蓝色。



小提示：

如需让HuskyLens学习或识别自己的脸，也就是自拍，此时看不到屏幕，那么可以根据RGB指示灯的不同颜色来确定状态。

2. 学习人脸：

将HuskyLens屏幕中央的“+”字对准需要学习的人脸，短按“学习按键”完成学习。如果识别到相同的脸，则屏幕上会出现一个蓝色的框并显示“人脸：ID1”。这说明已经可以进行人脸识别了。



然而，上述操作只是让HuskyLens学习了人脸的一个角度(维度)，但实际上人脸是立体的，因此人脸有多个角度。如果人脸角度变化了，比如正脸换成了侧脸，那HuskyLens不一定能识别出来。为解决这个问题，HuskyLens内置有持续学习的功能，能录入人脸的各个角度，让HuskyLens越学越准确。

录入各个角度人脸的操作方法如下: (注: 开始学习新的人脸之前, 先要让HuskyLens忘记已经学习过的人脸。操作方法参考“忘记学过的人脸”章节。)

将HuskyLens屏幕中央的“+”字对准需要学习的人脸, 长按“学习按键”不松开, 此时屏幕中会在人脸上显示黄色框并标识“人脸:ID1”, 说明HuskyLens正在学习人脸。然后, 将HuskyLens屏幕中央的黄色框依次对准同一个人的不同角度, 如正脸、侧脸(也可以是同一个人的多张照片), 录入此人脸的各个角度。学习过程中, RGB灯为黄色。

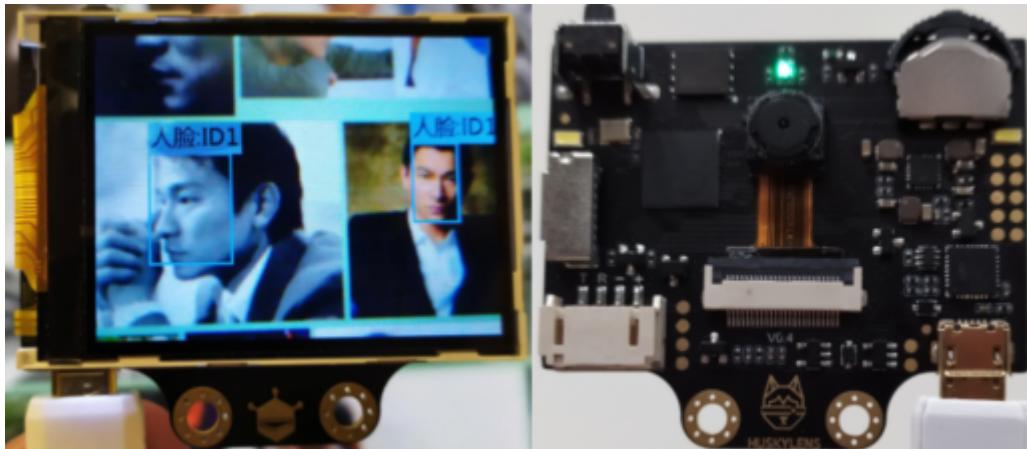


接着, 松开“学习按键”, 结束学习。如果HuskyLens识别到学习过的人脸, 则屏幕上, 这个人脸会被蓝色的框选中, 并显示“人脸: ID1”。

小提示: 如果屏幕中央没有“+”字, 说明HuskyLens在该功能下已经学习过了(已学习状态)。如要让HuskyLens学习新的人脸, 则需要让HuskyLens忘记已学的人脸。

3. 人脸识别:

HuskyLens学习过的人脸信息会自动保存起来。后续, 当HuskyLens检测到学习过的人脸时, 会将该人脸用蓝色方框框选出来并标识“人脸:ID1”, 边框大小会随着人脸大小而变化, 并自动追踪人脸。此时RGB指示灯为绿色。



4. 忘记学过的人脸:

如果屏幕中央没有“+”字, 说明HuskyLens在该功能下已经学习过了(已学习状态)。如要让HuskyLens学习新的人脸, 则需要删除之前学习过的人脸信息, 也就是让HuskyLens忘记已学的人脸。

删除已学东西的操作方法如下:

在当前功能下, 短按“学习按键”, 屏幕提示“再按一次遗忘!”。在倒计时结束前, 再次短按“学习按键”, 即可删除上次学习的东西, 屏幕中央显示“+”字, 说明HuskyLens已经准备

好学习新东西了。如果不小心短按了“学习按键”，屏幕已经提示“再按一次遗忘！”，但又不想删除已学习的东西，那么在倒计时结束前，不要有任何操作即可。



让HuskyLens忘记所学东西的操作方法，在其他功能下，也是完全一样的，后续不再复述。

识别多个人脸

默认设置为学习并识别单个人脸。如要学习并识别多个人脸，则需要在人脸识别功能的二级菜单参数设置中打开“学习多个”选项。

操作设置

1. 向左拨动“功能按键”，至屏幕顶部显示“人脸识别”。
2. 长按“功能按键”，进入人脸识别功能的二级菜单参数设置界面。
3. 向左或向右拨动“功能按键”，选中“学习多个”，然后短按“功能按键”，接着向右拨动“功能按键”打开“学习多个”的开关，即：进度条颜色变蓝，进度条上的方块位于进度条的右边。再短按“功能按键”，确认该参数。
4. 向左拨动“功能按键”，选中“保存并返回”，短按“功能按键”，屏幕提示“是否保存参数？”。默认选择“确认”，此时短按“功能按键”，即可保存参数，并自动返回到人脸识别模式。



学习与识别

1. 学习多个人脸：

将HuskyLens屏幕中央的“+”字对准需要学习的人脸，长按“学习按键”完成第一个人脸的学习（各个角度）。松开“学习按键”后，屏幕上会提示：“再按一次按键继续！按其他按键结束”。如要继续学习下一个人脸，则在倒计时结束前短按“学习按键”，可以继续学习下一个人脸。如果不再需要学习其他人脸了，则在倒计时结束前短按“功能按键”即可，或者不操作任何按键，等待倒计时结束。

本章节中，我们需要继续学习下一个人脸，因此在倒计时结束前短按“学习按键”。然后将HuskyLens屏幕中央的“+”字对准需要学习的下一个人脸，长按“学习按键”完成第二个人脸的学习。以此类推。

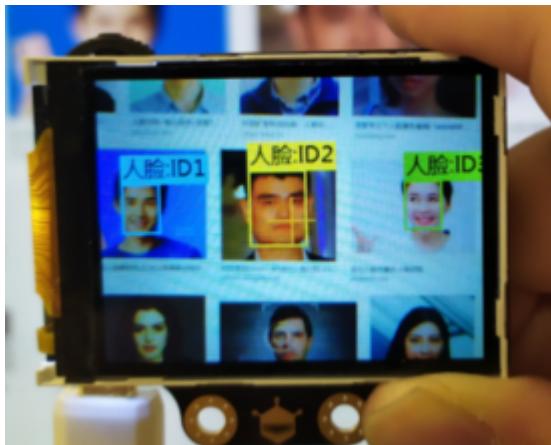
HuskyLens标注的人脸ID与录入人脸的先后顺序是一致的，也就是：学习过的人脸会按顺序依次标注为“人脸：ID1”，“人脸：ID2”，“人脸：ID3”，以此类推，并且不同的人脸ID对应的边框颜色也不同。



小提示：如果屏幕中央没有“+”字，说明HuskyLens在该功能下已经学习过了（已学习状态）。如要让HuskyLens学习新人脸，则需要让HuskyLens忘记已学的。操作方法参考“忘记学过的人脸”章节。

2. 识别多个人脸：

HuskyLens学习过的人脸信息会自动保存起来。后续，当HuskyLens检测到学习过的人脸时，会将这些人脸用方框框选出来并标识ID，第一个学习的人脸标注为“人脸：ID1”，第二个学习的人脸标注为“人脸：ID2”，第三个学习的人脸标注为“人脸：ID3”，以此类推。不同的人脸ID对应的边框颜色也不同，边框大小会随着人脸大小而变化，并自动追踪人脸。

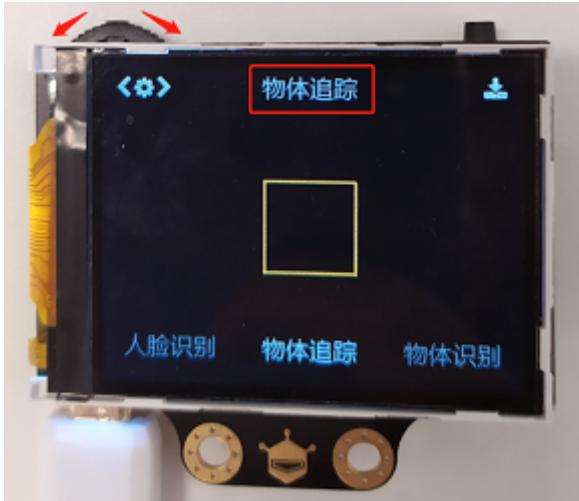


物体追踪

本功能可以学习并追踪一个指定的物体。只能追踪一个物体，暂不支持追踪多个物体。

操作设置

1. 向左或向右拨动“功能按键”，至屏幕顶部显示“物体追踪”。



2. 长按“功能按键”，进入物体追踪功能的二级菜单参数设置界面。
3. 向右拨动“功能按键”，选中“学习开启”，然后短按“功能按键”，接着向右拨动“功能按键”打开“学习开启”的开关，即：进度条颜色变蓝，进度条上的方块位于进度条的右边。再短按“功能按键”，确认该参数。
4. 同样的方法，打开“自动保存模型”的开关。

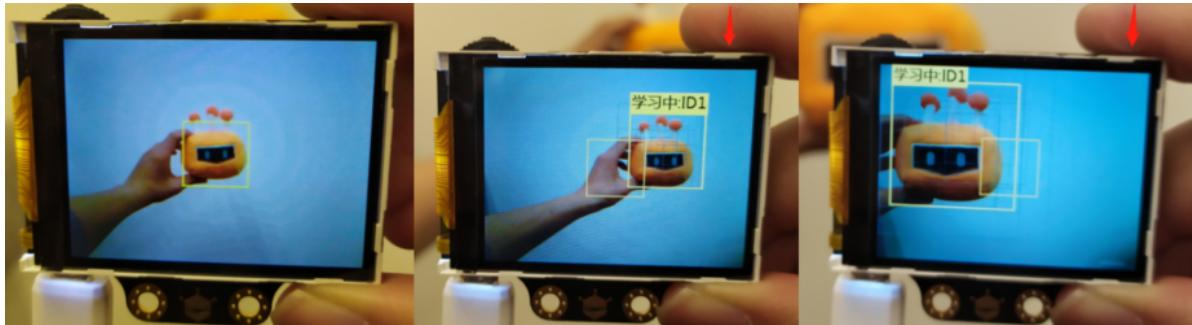


5. 也可以设置“学习框长宽比”和“学习框大小”来调整学习框的尺寸，以便于更好的匹配所学物体的尺寸。
6. 向左拨动“功能按键”，选中“保存并返回”，短按“功能按键”，屏幕提示“是否保存参数？”，默认选择“确认”，此时短按“功能按键”，即可保存参数，并自动返回到物体追踪模式。

学习与追踪

1. 学习物体：

把HuskyLens对准需要追踪的物体，调节物体与HuskyLens的距离，将物体包含在屏幕中央的橙黄色方框内。如不方便，包含特征鲜明的局部亦可。长按“学习按键”不松开，并调整角度和距离，使得HuskyLens从不同的角度和距离学习该物体。学习过程中，屏幕上的黄框会标注“学习中：ID1”。

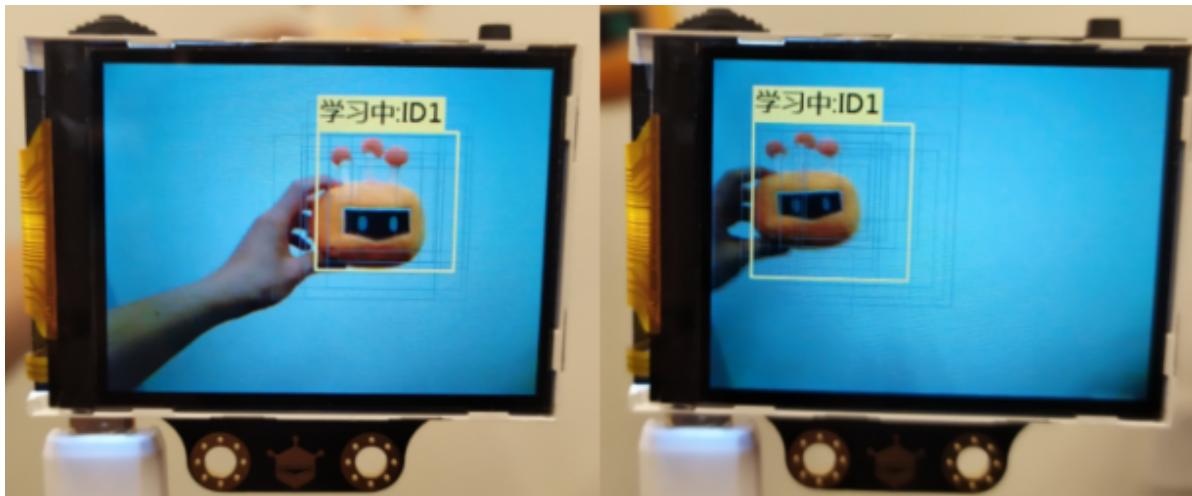


当HuskyLens在不同的角度和距离都能追踪到该物体时，就可以松开“学习按键”结束学习了。

小提示：如果屏幕中央没有橙黄色方框，说明HuskyLens之前学习过一个物体，请参照“忘记学过的人脸”章节进行操作。

2. 追踪物体：

移动HuskyLens或者目标物体，屏幕中的框会自动追踪目标物体。追踪物体时，会显示“学习中：ID1”，表示HuskyLens一边追踪物体，一边学习，这样设置有助于提高物体追踪的能力；也可以长按“功能按键”，进入二级菜单参数设置，选择“学习开启”，关闭此参数。



当识别结果满足要求，达到预期效果，就可关闭一边追踪一边学习的功能，方法是：长按“功能按键”，进入物体追踪功能的二级菜单参数设置，选择“学习开启”，关闭此参数即可（进度条颜色变白，进度条上的方块位于进度条的左边）。

小提示：

追踪物体时，每次只能追踪一个物体，可以是任何有明显轮廓的物体，甚至是各种手势。

物体识别

本功能可识别这是什么物体，并追踪。目前仅支持20种物体，分别为：飞机、自行车、鸟、船、瓶子、巴士、汽车、猫、椅子、牛、餐桌、狗、马、摩托车、人、盆栽植物、羊、沙发、火车、电视。

20种物体的英文名称，分别为： aeroplane, bicycle, bird, boat, bottle, bus, car, cat, chair, cow, diningtable, dog, horse, motorbike, person, pottedplant, sheep, sofa, train, tvmonitor.

默认设置为只标记并识别一个物体。本章节采用标记并识别多个物体为例进行演示。

操作设置

1. 向左或向右拨动“功能按键”，直至屏幕顶部显示“物体识别”。
2. 长按“功能按键”，进入物体识别功能的二级菜单参数设置界面。
3. 向左或向右拨动“功能按键”，选中“学习多个”，然后短按“功能按键”，接着向右拨动“功能按键”打开“学习多个”的开关，即：进度条颜色变蓝，进度条上的方块位于进度条的右边。再短按“功能按键”，确认该参数。



4. 向左拨动“功能按键”，选中“保存并返回”，短按“功能按键”，屏幕提示“是否保存参数？”，默认选择“确认”，此时短按“功能按键”，即可保存参数，并自动返回到物体识别模式。

标记与识别

1. 借测物体：

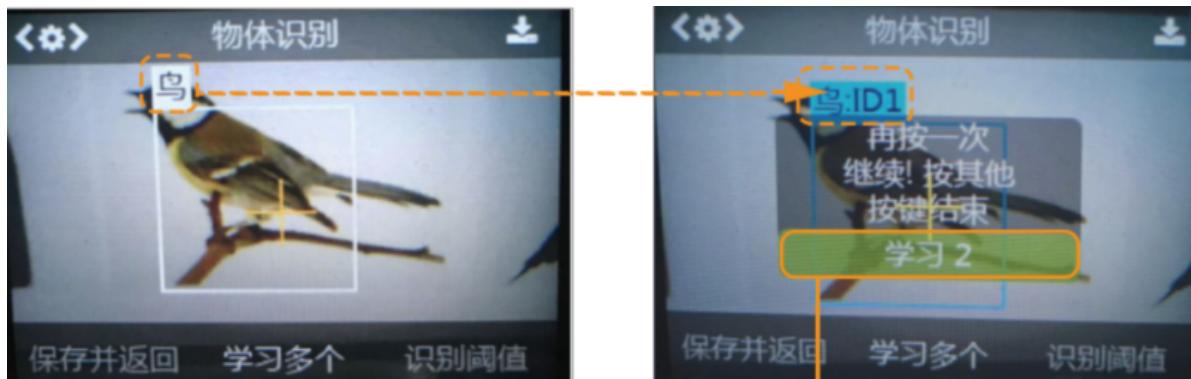
把HuskyLens对准目标物体，在屏幕上，会有白色框自动框选出识别到的所有物体，并显示对应的物体名称。目前只能识别并框选20种物体，其余物体无法识别和框选。



2. 标记物体：

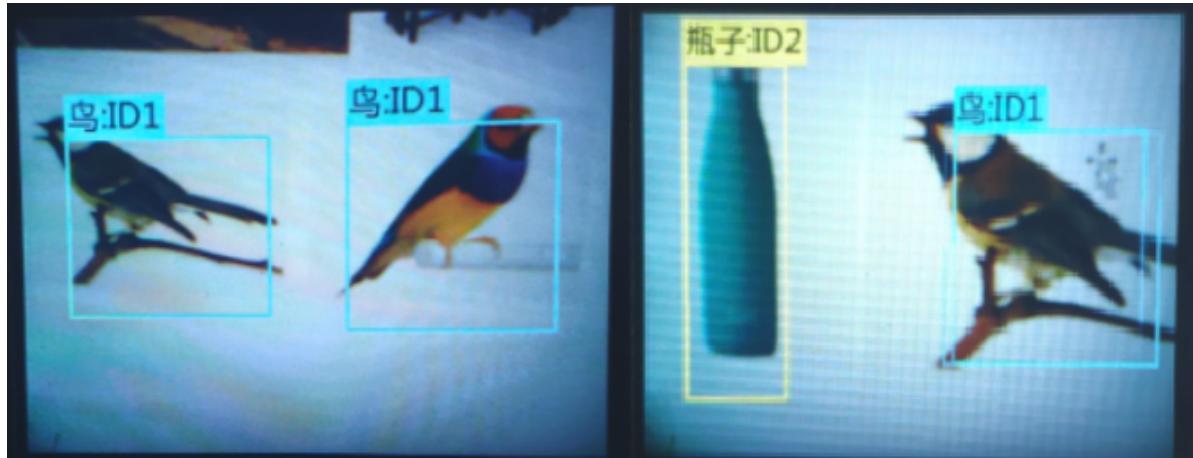
把HuskyLens对准目标物体，当屏幕上显示的物体被检测到并显示其名字时，将屏幕中央的“+”字对准该物体的白色框中央，短按“学习按键”进行标记。此时，框体颜色由白色变为蓝色，并显示其名字和ID1，同时有消息提示：“再按一次继续，按其他按键结束”。如要继续标记下一个物体，则在倒计时结束前按下“学习按键”，可以继续标记下一个物体。如果不再需要标记其他物体了，则在倒计时结束前按下“功能按键”即可，或者不操作任何按键，等待倒计时结束。

HuskyLens显示的物体ID与标记物体的先后顺序是一致的，也就是：ID会按顺序依次标注为“ID1”，“ID2”，“ID3”，以此类推，并且不同的物体ID对应的边框颜色也不同。



3. 识别物体：

HuskyLens再次遇到标记过的物体时，在屏幕上会有彩色的边框框选出这些物体，并显示物体名称与ID。边框的大小随着物体的大小而变化，自动追踪这些物体。同类物体，有相同颜色的边框、名字和ID。支持同时识别多类物体，比如同时识别出瓶子和鸟。



这个功能，可以作为一个简单的筛选器，从一堆物体中找出你需要的物体，并做追踪。

小提示：

此功能不能区分同类物体间的不同，比如：只能识别出这是猫，但不能识别出这是什么猫。有别于人脸识别，人是一类，但可以区分不同的人脸。

巡线追踪

本功能可以追踪指定颜色的线条，做路径预测。默认设置为只追踪一种颜色的线条。本章节以只追踪一种颜色的线条为例进行说明。

操作设置

1. 向左或向右拨动“功能按键”，直至屏幕顶部显示“巡线”。
2. 长按“功能按键”，进入巡线功能的二级菜单参数设置界面。
3. 向左或向右拨动“功能按键”，选中“学习多个”，然后短按“功能按键”，查看“学习多个”的开关是否处于关闭状态。如果没关，就向左拨动“功能按键”关闭“学习多个”的开关，即：进度条颜色变白，进度条上的方块位于进度条的左边。再短按“功能按键”，确认该参数。
4. 如果环境光线比较暗，可以打开补光灯。参照上述方法，将“LED开关”打开即可。

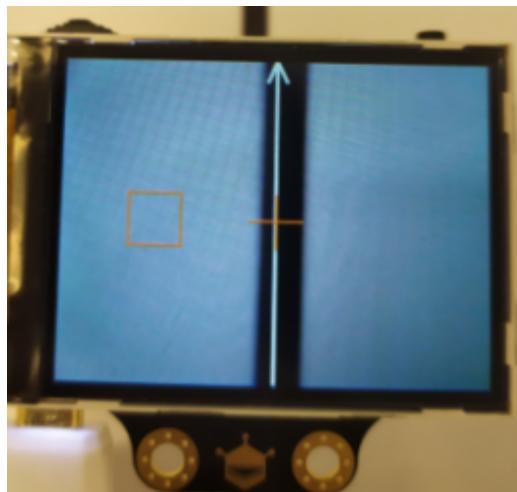


5. 向左拨动“功能按键”，选中“保存并返回”，短按“功能按键”，屏幕提示“是否保存参数？”，默认选择“确认”，此时短按“功能按键”，即可保存参数，并自动返回到巡线模式。

学习与追踪

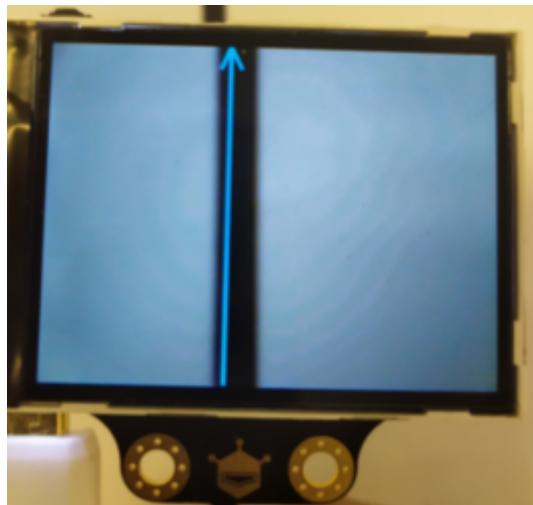
1. 学习线条：

将HuskyLens屏幕上的“+”字对准目标线条，将橙黄色的方框对准背景色。建议HuskyLens的视野范围内只有需要学习的线条，并且没有交叉线。尽量将HuskyLens与目标线条保持平行，然后HuskyLens会自动检测线条，并出现白色的箭头。然后短按“学习按键”即可，白色箭头变成了蓝色箭头。



2. 巡线追踪：

当HuskyLens检测到学习过的线条时（即：同一种颜色的线条），HuskyLens的屏幕上会显示蓝色的箭头，箭头的指向表示路径预测的方向。



小提示：

- 学习线条时，尽量把HuskyLens的位置调节到与线条平行。
- 识别的线条可以是任何与背景有明显色差的单色线条，建议线条与背景色有足够的色差，以保证巡线的稳定。

- HuskyLens可以根据线条的颜色不同，进行多个线条的学习与巡线，但是线条必须是与背景有明显色差的单色线条。大多数情况下，巡线用的线条，只有一种颜色。为了保证稳定性，建议只对一种颜色的线条进行巡线。
- 线条的颜色与环境光线有很大关系，建议巡线的时候，保持环境光线的稳定。

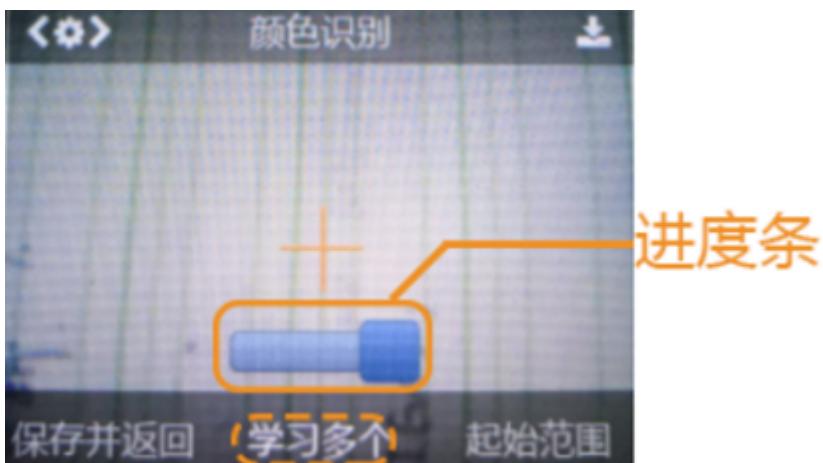
颜色识别

本功能可以学习、识别、追踪指定的颜色。

默认设置为只学习、识别并追踪一种颜色。本章节采用学习、识别并追踪多种颜色为例进行说明。

操作设置

1. 向左或向右拨动“功能按键”，直至屏幕顶部显示“颜色识别”。
2. 长按“功能按键”，进入颜色识别功能的二级菜单参数设置界面。
3. 向左或向右拨动“功能按键”，选中“学习多个”，然后短按“功能按键”，接着向右拨动“功能按键”打开“学习多个”的开关，即：进度条颜色变蓝，进度条上的方块位于进度条的右边。再短按“功能按键”，确认该参数。

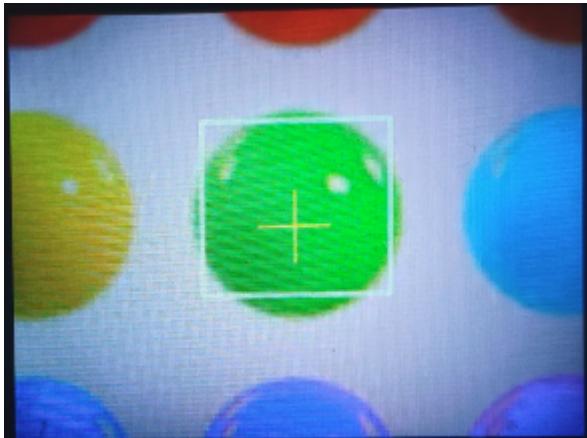


4. 向左拨动“功能按键”，选中“保存并返回”，短按“功能按键”，屏幕提示“是否保存参数？”，默认选择“确认”，此时短按“功能按键”，即可保存参数，并自动返回到颜色识别模式。

学习与识别

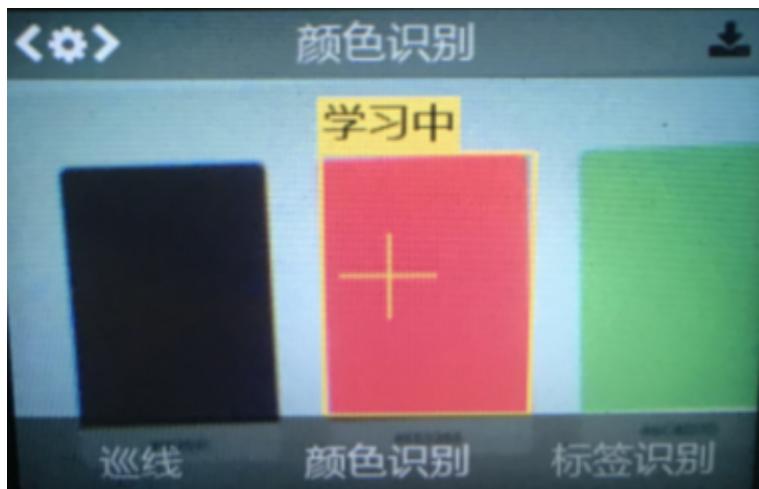
1. 借测颜色

将HuskyLens屏幕中央的“+”字对准目标颜色块，屏幕上会有一个白色方框，自动框选目标颜色块。调整HuskyLens与颜色块的角度和距离，让白色方框尽量框住整个目标色块。



2. 学习颜色

侦测到颜色后，按下“学习按键”学习第一种颜色，然后松开“学习按键”结束学习，屏幕上有消息提示：“再按一次继续，按其他按键结束”。如要继续学习下一种颜色，则在倒计时结束前按下“学习按键”，可以继续学习下一种颜色。如果不再需要学习其他颜色了，则在倒计时结束前按下“功能按键”即可，或者不操作任何按键，等待倒计时结束。HuskyLens显示的颜色ID与学习颜色的先后顺序是一致的，也就是：ID会按顺序依次标注为“ID1”，“ID2”，“ID3”，以此类推，并且不同颜色对应的边框颜色也不同。

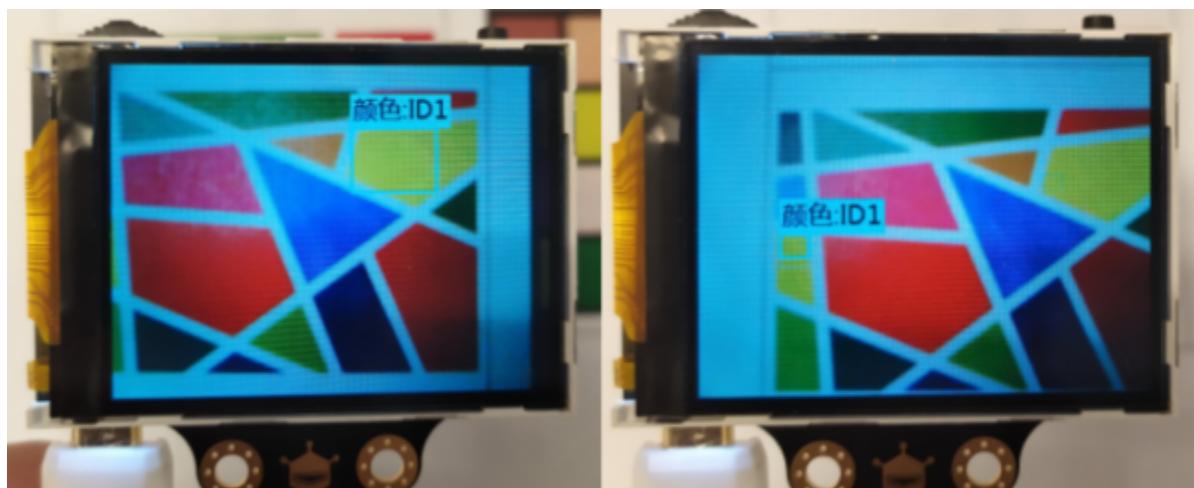


3. 识别颜色

如HuskyLens遇到相同或近似的颜色，屏幕上会有彩色边框框选出色块，并显示该颜色的ID，边框的大小随颜色块的面积一起变化，边框会自动跟踪色块。多种不同的颜色可以同时识别并追踪，不同颜色对应的边框颜色也不同。



当出现多个相同颜色的色块时，相隔的色块不能被同时识别，只能一次识别一个色块。



小提示：环境光线对颜色识别的影响很大，对于相近的颜色，HuskyLens有时会误识别。建议保持环境光线的稳定，在光线适中的环境中使用此功能。

标签识别

本功能可以侦测二维码标签；学习、识别、追踪指定的二维码标签。

默认设置为只学习、识别并追踪一个二维码标签。本章节采用学习、识别并追踪多个二维码标签为例进行说明。

操作设置

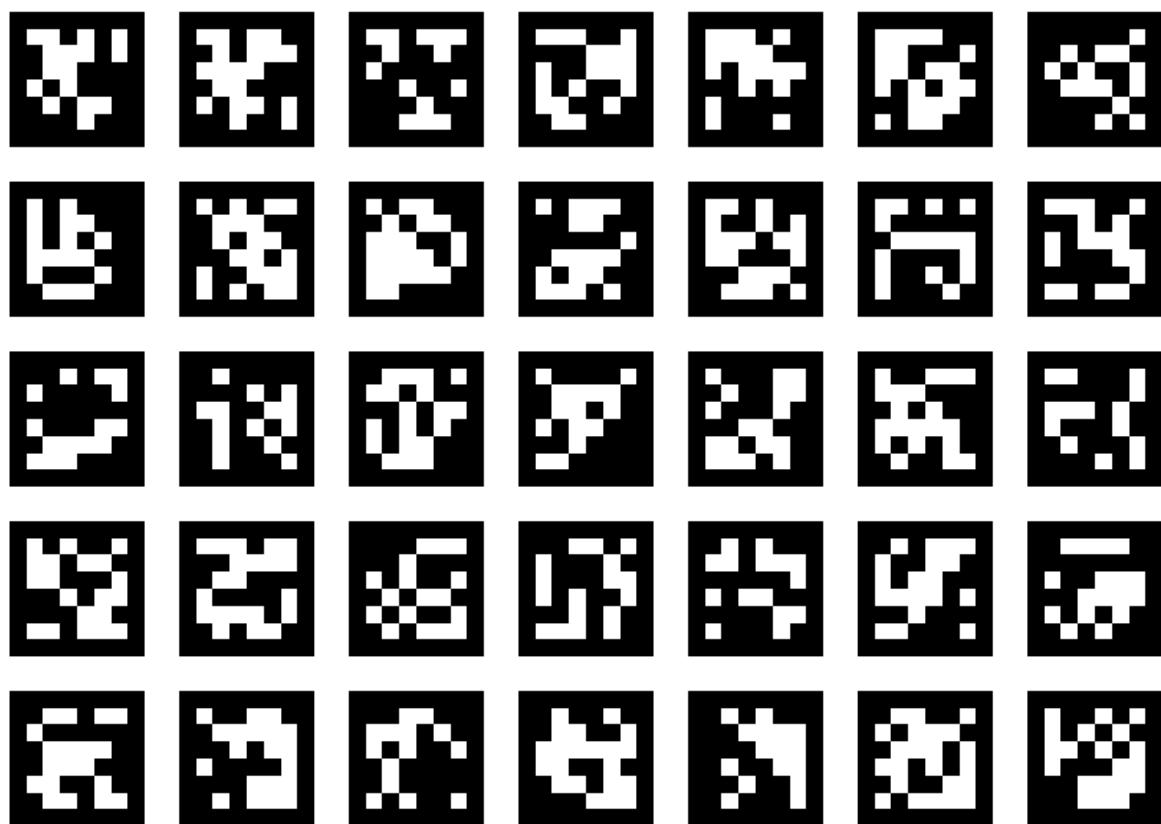
1. 向左或向右拨动“功能按键”，直至屏幕顶部显示“标签识别”。
2. 长按“功能按键”，进入标签识别功能的二级菜单参数设置界面。
3. 向左或向右拨动“功能按键”，选中“学习多个”，然后短按“功能按键”，接着向右拨动“功能按键”打开“学习多个”的开关，即：进度条颜色变蓝，进度条上的方块位于进度条的右边。再短按“功能按键”，确认该参数。



4. 向左拨动“功能按键”，选中“保存并返回”，短按“功能按键”，屏幕提示“是否保存参数？”，默认选择“确认”，此时短按“功能按键”，即可保存参数，并自动返回到标签识别模式。

学习与识别

可用下面的二维码标签来测试。



1. 检测标签

当HuskyLens检测到二维码标签时，屏幕上会用白色框自动框选出检测到的所有二维码标签。



2. 学习标签

将HuskyLens屏幕中央的“+”字对准需要学习的标签，短按或长按“学习按键”完成第一个标签的学习。松开“学习按键”后，屏幕上会提示：“再按一次按键继续！按其他按键结束”。如要继续学习下一个标签，则在倒计时结束前按下“学习按键”，可以继续学习下一个标签。如果不再需要学习其他标签了，则在倒计时结束前按下“功能按键”即可，或者不操作任何按键，等待倒计时结束。



本章节中，需要继续学习下一个标签，因此在倒计时结束前按下“学习按键”，然后将 HuskyLens屏幕中央的“+”字对准需要学习的下一个标签，短按或长按“学习按键”完成第二个标签的学习。以此类推。

标签ID与录入标签先后顺序是一致的，也就是：学习过的标签会按顺序依次标注为“标签：ID1”，“标签：ID2”，“标签：ID3”，以此类推，并且不同的标签对应的边框颜色也不同。

3. 识别标签

HuskyLens再次遇到学习过的标签时，在屏幕上会有彩色的边框框选出这些标签，并显示其ID。边框的大小会随着二维码标签的大小进行变化，边框自动追踪这些二维码标签。



micro:bit教程

本章节使用MakeCode网页版和Mind+软件，分别用几个案例演示具体的用法。

MakeCode加载HUSKYLENS插件

1. 在[MakeCode网页版](#)中新建一个项目，然后，点击右上方的“更多”按钮（齿轮图标），在其下拉菜单中选择“扩展”，打开扩展界面。



2. 在搜索框中输入<https://github.com/tangjie133/pxt-huskylens>，然后点击搜索按钮（搜索框右边的放大镜按钮），就能看到HuskyLens插件，然后点击它，就能把HuskyLens的插件加载进来了。



3. 编程界面中，你就能看到Huskylens模块，点击它，就会弹出指令积木块。



MakeCode模块说明

| 模块 | 说明 |
|----|--|
| | 初始化。只需要执行一次，放在主程序开始位置。需要将HuskyLens的接口输出协议设置为I2C或自动。初始化成功为“√”错误为“×” |
| | 切换算法(功能)。选择你需要的算法，暂时只能选择一个。切换算法需要一段时间，切换成功，切换成功为笑脸，错误则反之。 |
| | micro:bit向HuskyLens请求一次数据，并存入“结果”（存储在micro:bit的内存变量中，一次请求刷新一次存储在内存中的数据），之后可以从“结果中获取数据”。只有调用此模块之后才会得到最新的数据。 |
| | 从请求得到的“结果”中获取IDx是否已经学习。 |
| | 从请求得到的“结果”中获取IDx的方框或箭头是否在画面中。 |

| 模块 | 说明 |
|----|---|
| | 从请求得到的“结果”中获取IDx方框的参数, 有: 方框的中心x坐标、中心Y坐标、长度、宽度。 |
| | 从请求得到的“结果”中获取IDx箭头的参数, 有: X起点、Y起点、X终点、Y终点。 |

micro:bit案例1：人脸识别

本章节演示如何把HuskyLens连接到micro:bit主板，然后micro:bit主板从HuskyLens读取人脸识别结果。如果HuskyLens认出主人（学习过的人脸），则micro:bit的点阵屏显示笑脸，否则显示哭脸。

HuskyLens与micro:bit的通信接口采用I2C接口。

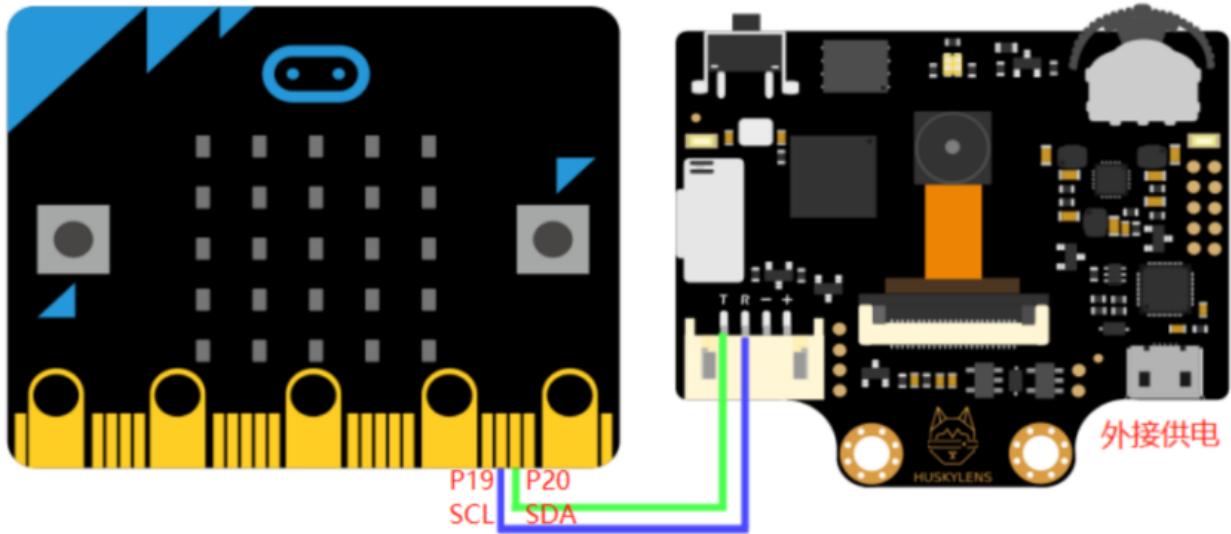
准备材料

- 硬件
 - [micro:bit主板](#) x 1
 - [micro:bit扩展板](#) x1
 - [HUSKYLENS](#) x 1
 - 4Pin连接线（或杜邦线）
- 软件
 - [Microsoft MakeCode for micro:bit](#)
 - [HUSKYLENS MakeCode插件](#)

硬件连线图

下图只作为连线示意图，HuskyLens的R和T管脚（此处其功能则为SCL与SDA）分别连接到micro:bit的SCL（P19）与SDA（P20）管脚，说明HuskyLens与micro:bit的通信接口采用I2C接口。

推荐采用[micro:bit扩展板](#)，简化连线。



小提示：

HuskyLens消耗的电流比较多，可达3.3V 320mA以上，单靠micro:bit主板供电是不够的，因此需要外接供电，可通过micro:bit扩展板的外接供电接口或者HuskyLens的USB接口实现。

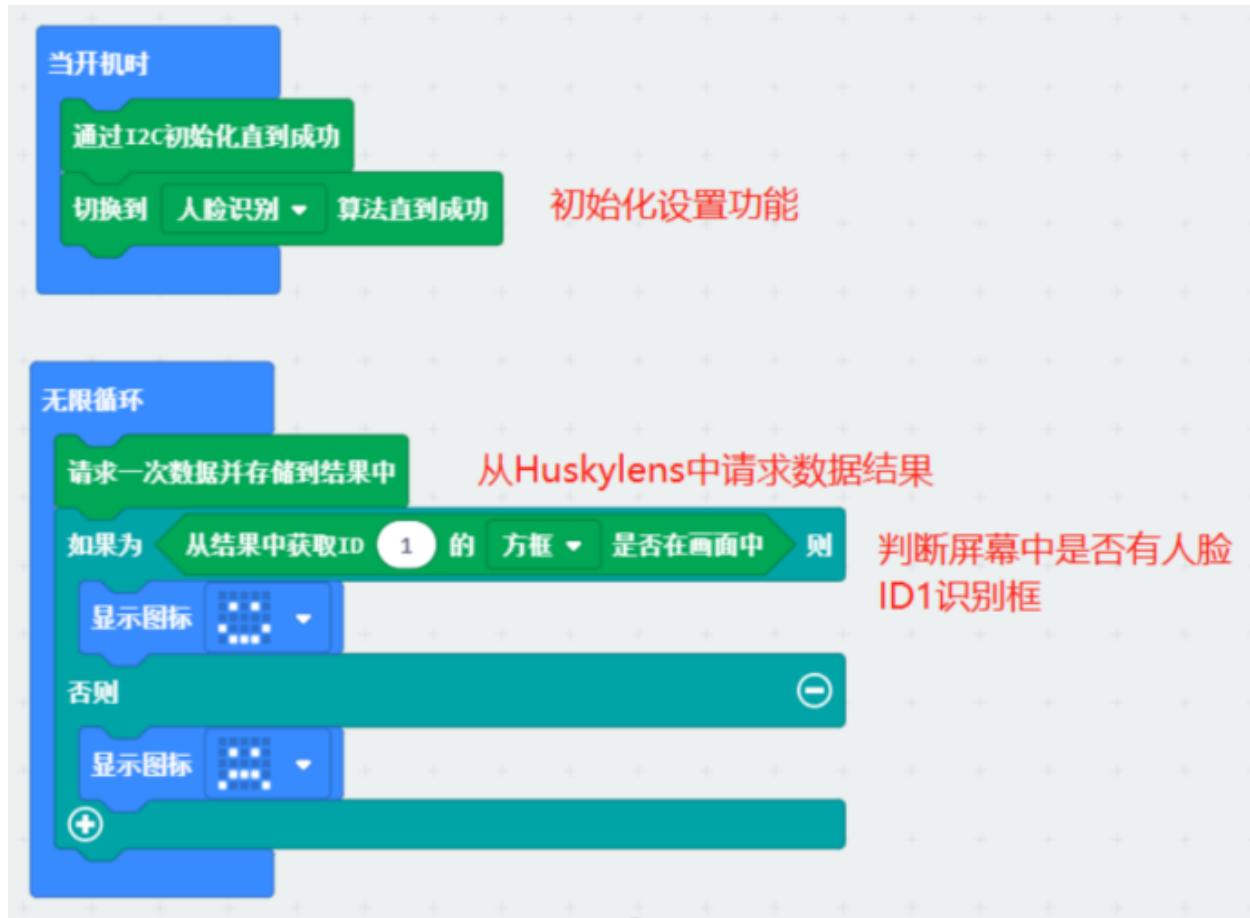
HUSKYLENS常规设置

把HuskyLens的通讯模式设置为I2C。



也可以设置为“自动识别模式”，无需反复调节通讯协议，简单方便。

样例代码



预期结果

1. 上传上述代码至micro:bit主板。
2. 参考前面讲解人脸识别功能的章节，让Huskylens学习完一个人脸，比如你的脸。
3. 当Huskylens识别到你的脸后，micro: bit主板上的点阵屏会显示笑脸，如果不是你的脸，或者没有人脸出现，则显示哭脸。

micro:bit案例2

更多案例，敬请期待。

Mind+案例

Mind+有Huskylens专题教程，请[点击此处](#)跳转到该教程。此处不再赘述。

Arduino教程

本章节使用Arduino IDE，分别用几个案例演示具体的用法。

安装Arduino库的一些注意点：

1. 解压下载的库文件，将“HUSYLENS”文件夹复制粘贴至Arduino IDE所在的“libraries”文件夹下。

打开Arduino IDE程序所在位置

| 名称 | 修改日期 | 类型 | 大小 |
|-----------------------|-------------------------|------------|-----------|
| drivers | 2019/10/29 17:22 | 文件夹 | |
| examples | 2019/10/29 17:22 | 文件夹 | |
| hardware | 2019/10/29 17:22 | 文件夹 | |
| java | 2019/10/29 17:22 | 文件夹 | |
| lib | 2019/10/29 17:22 | 文件夹 | |
| libraries | 2019/11/29 14:32 | 文件夹 | |
| reference | 2019/10/29 17:22 | 文件夹 | |
| tools | 2019/10/29 17:22 | 文件夹 | |
| tools-builder | 2019/10/29 17:22 | 文件夹 | |
| arduino.exe | 2019/9/13 18:23 | 应用程序 | 395 KB |
| arduino.l4j.ini | 2019/9/13 18:23 | 配置设置 | 1 KB |
| arduino_debug.exe | 2019/9/13 18:23 | 应用程序 | 393 KB |
| arduino_debug.l4j.ini | 2019/9/13 18:23 | 配置设置 | 1 KB |
| arduino-builder.exe | 2019/9/13 18:23 | 应用程序 | 14,321 KB |
| libusb0.dll | 2019/9/13 18:23 | 应用程序扩展 | 43 KB |
| msvcp100.dll | 2019/9/13 18:23 | 应用程序扩展 | 412 KB |
| msvcr100.dll | 2019/9/13 18:23 | 应用程序扩展 | 753 KB |
| revisions.txt | 2019/9/13 18:23 | 文本文档 | 90 KB |
| uninstall.exe | 2019/10/29 17:24 | 应用程序 | 404 KB |
| wrapper-manifest.xml | 2019/9/13 18:23 | XML 文档 | 1 KB |

打开库文件

复制粘贴文件

| 名称 | 修改日期 | 类型 | 大小 |
|-----------------------------|-------------------------|------------|------|
| Adafruit_Circuit_Playground | 2019/10/29 17:22 | 文件夹 | |
| Bridge | 2019/10/29 17:22 | 文件夹 | |
| Esplora | 2019/10/29 17:22 | 文件夹 | |
| Ethernet | 2019/10/29 17:22 | 文件夹 | |
| Firmata | 2019/10/29 17:22 | 文件夹 | |
| GSM | 2019/10/29 17:22 | 文件夹 | |
| HUSYLENS | 2019/12/16 11:34 | 文件夹 | |
| Keyboard | 2019/10/29 17:22 | 文件夹 | |
| LiquidCrystal | 2019/10/29 17:22 | 文件夹 | |
| Mouse | 2019/10/29 17:22 | 文件夹 | |
| Robot_Control | 2019/10/29 17:22 | 文件夹 | |
| Robot_Motor | 2019/10/29 17:22 | 文件夹 | |
| RobotIRremote | 2019/10/29 17:22 | 文件夹 | |
| SD | 2019/10/29 17:22 | 文件夹 | |
| Servo | 2019/10/29 17:22 | 文件夹 | |
| SpacebrewYun | 2019/10/29 17:22 | 文件夹 | |
| Stepper | 2019/10/29 17:22 | 文件夹 | |
| Temboo | 2019/10/29 17:22 | 文件夹 | |
| TFT | 2019/10/29 17:22 | 文件夹 | |
| WiFi | 2019/10/29 17:22 | 文件夹 | |
| .keep | 2019/9/13 18:23 | KEEP 文件 | 0 KB |

2. 库文件夹的名字必须为“HUSYLENS”。

3. 所有.h和.cpp文件必须放在库文件夹的根目录下，绝不可以放在二级目录或文件夹中。

| 本地磁盘 (D:) > arduino-1.8.2 > libraries > HUSKYLENS | | | |
|---|-----------------|--------|-------|
| 名称 | 修改日期 | 类型 | 大小 |
| .vscode | 2020/1/23 12:11 | 文件夹 | |
| examples | 2020/1/23 12:11 | 文件夹 | |
| DFMobile.cpp | 2020/1/23 12:11 | CPP 文件 | 2 KB |
| DFMobile.h | 2020/1/23 12:11 | H 文件 | 1 KB |
| HUKYLENS.cpp | 2020/1/23 12:11 | CPP 文件 | 1 KB |
| HUKYLENS.h | 2020/1/23 12:11 | H 文件 | 19 KB |
| HUKYLENSMindPlus.cpp | 2020/1/23 12:11 | CPP 文件 | 1 KB |
| HUKYLENSMindPlus.h | 2020/1/23 12:11 | H 文件 | 7 KB |
| HuskyLensProtocolCore.c | 2020/1/23 12:11 | C 文件 | 6 KB |
| HuskyLensProtocolCore.h | 2020/1/23 12:11 | H 文件 | 1 KB |
| PIDLoop.h | 2020/1/23 12:11 | H 文件 | 3 KB |

Arduino案例1：串口打印物体坐标数据

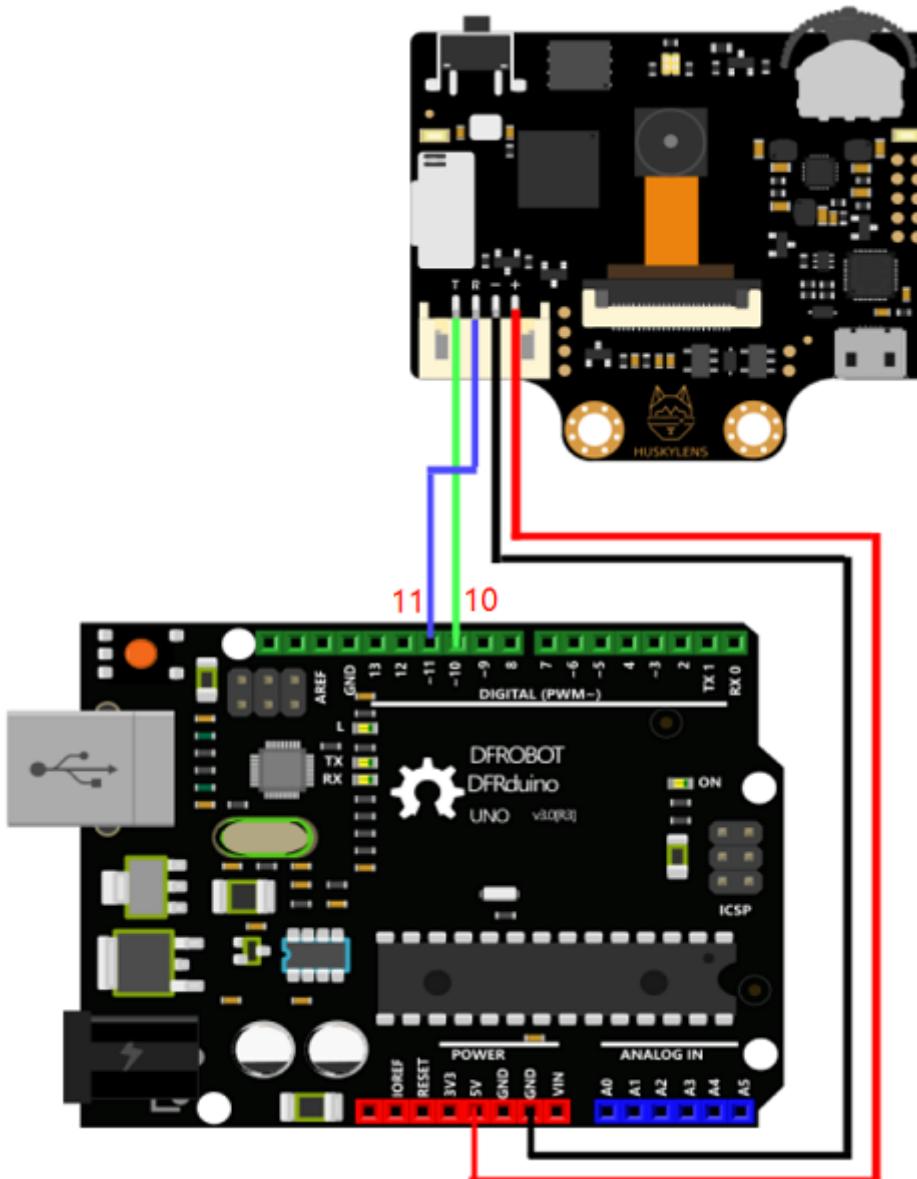
本章节演示如何把HuskyLens连接到Arduino主板，然后Arduino主板从HuskyLens读取数据，并将读到的数据在串口监视器中打印出来。分UART模式和I2C模式，分别演示。

准备材料

- 硬件
 - [DFRduino UNO R3](#) (或类似主板) x 1
 - [HUSKYLENS](#) x 1
 - 4pin连接线 (或杜邦线)
- 软件
 - [Arduino IDE](#)(推荐1.8.x版本)
 - [下载并安装 HUSKYLENS Library。如何安装库？](#)

UART模式

硬件连线图



HUSKYLENS常规设置

把HuskyLens的通讯模式设置为UART，波特率9600。



也可以设置为“自动识别模式”，无需反复调节通讯协议，简单方便。

样例代码

```
#include "HUKSYLENS.h"
#include "SoftwareSerial.h"

HUKSYLENS huskylens;
SoftwareSerial mySerial(10, 11); // 设置软串口引脚 RX >> Pin 10, TX >> Pin 11
//HUKSYLENS green line >> Pin 10; blue Line >> Pin 11

void setup() {
    Serial.begin(115200); // 初始化硬串口 (与计算机通信)
    mySerial.begin(9600); // 初始化软串口 (与huskylens通信, 与huskylens 常规设置 >> 协议种类
    //中串口波特率一致)
    while (!huskylens.begin(mySerial)) // 与huskylens通信失败
    {
        Serial.println(F("Begin failed!"));
        Serial.println(F("1.Please recheck the \"Protocol Type\" in HUKSYLENS (General
        Settings>>Protocol Type>>Serial 9600)"));
        Serial.println(F("2.Please recheck the connection."));
        delay(500);
    }
}

void loop() {
    if (!huskylens.request()) Serial.println(F("Fail to request objects from
    HUKSYLENS!")); // 未从huskylens中请求对象信息
    else if(!huskylens.isLearned()) Serial.println(F("Object not learned!")); // 
    huskylens处于待学习状态
    else if(!huskylens.available()) Serial.println(F("Object disappeared!")); // 
    huskylens未识别到学习过的对象
    else
    {
        HUKSYLENSResult result = huskylens.read();
        if (result.command == COMMAND_RETURN_BLOCK)
        {

Serial.println(String()+F("Block:xCenter=")+result.xCenter+F(",yCenter=")+result.yCenter+
F(",width=")+result.width+F(",height=")+result.height); // 返回huskylens识别框的中心坐标和
宽度
        }
        else if (result.command == COMMAND_RETURN_ARROW)
        {

Serial.println(String()+F("Arrow:xOrigin=")+result.xOrigin+F(",yOrigin=")+result.yOrigin+
F(",xTarget=")+result.xTarget+F(",yTarget=")+result.yTarget); // 返回huskylens识别到的线条
矢量的起点坐标与终点坐标
        }
        else
        {
            Serial.println("Object unknown!"); // huskylens检测到未学习过的对象
        }
    }
}
```

预期结果

1. 上传上述代码至Arduino主板
2. 参考前面讲解各个功能的章节，让HuskyLens学习完一个新东西，比如人脸。
3. 打开Arduino IDE的串口监视器查看结果。

当HuskyLens处于人脸识别、物体追踪、物体识别、颜色识别、标签识别模式时，则打印的结果如下图：

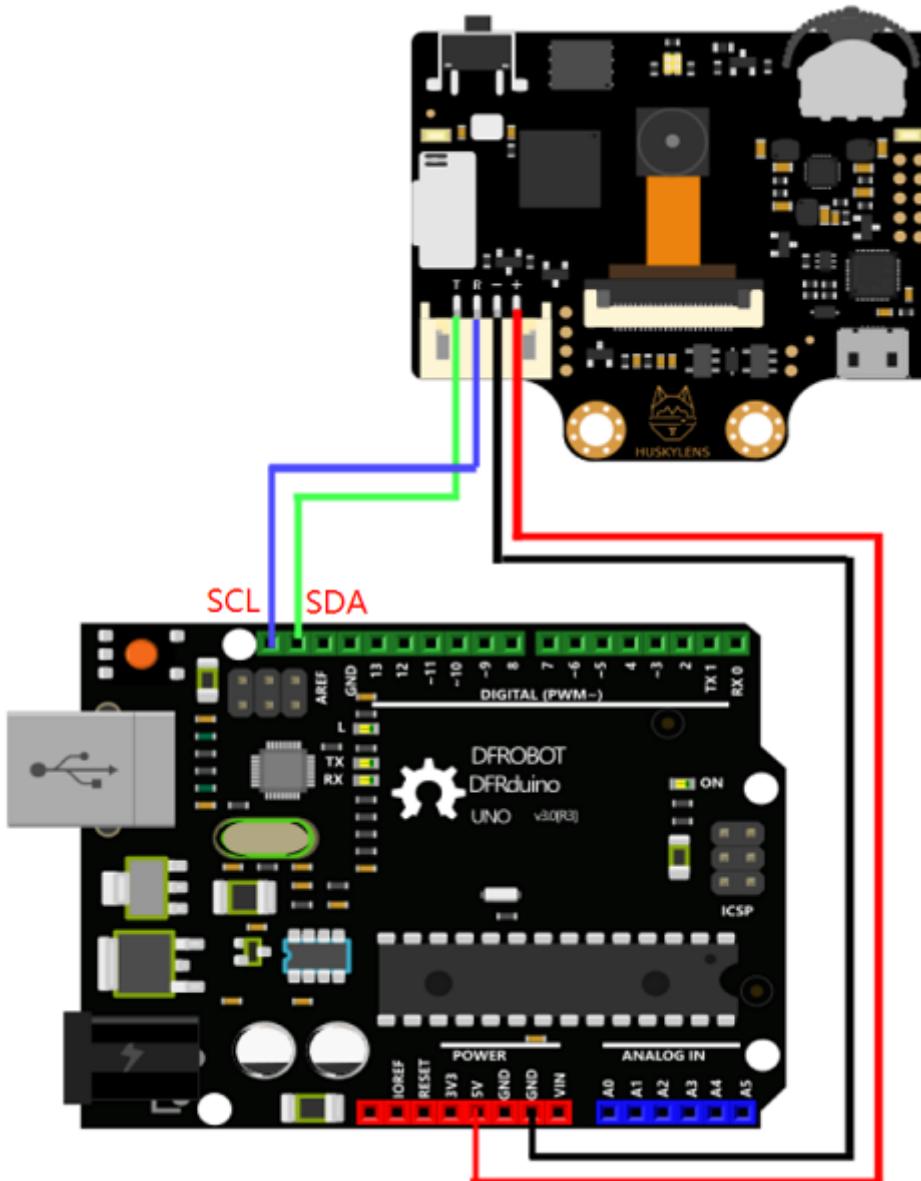
```
Block:xCenter=162,yCenter=135,width=138,height=146
Block:xCenter=163,yCenter=134,width=138,height=146
Block:xCenter=163,yCenter=134 width=138,height=146 ← 返回识别框的宽高
Block:xCenter=162,yCenter=137,width=108,height=145
Block:xCenter=162,yCenter=135,width=108,height=146
Block:xCenter=162,yCenter=135,width=108,height=146
Block:xCenter=162,yCenter=135,width=108,height=146
Block:xCenter=163,yCenter=135,width=138,height=146
Block:xCenter=163,yCenter=136,width=109,height=146
Block:xCenter=163,yCenter=136,width=109,height=146
Block:xCenter=162,yCenter=137,width=138,height=145
Block:xCenter=162,yCenter=136,width=138,height=146
Block:xCenter=162,yCenter=136,width=138,height=146
Block:xCenter=162,yCenter=137,width=108,height=145
Block:xCenter=162,yCenter=137 width=108,height=145 ← 返回识别框的中心坐标
Block:xCenter=163,yCenter=135,width=109,height=146
Block:xCenter=163,yCenter=137,width=109,height=145
```

当HuskyLens处于巡线追踪模式时，则打印的结果如下图：

```
Arrow:xOrigin=304,yOrigin=158,xTarget=216,yTarget=82 ← 线条矢量起点坐标
Arrow:xOrigin=312,yOrigin=164,xTarget=224,yTarget=82
Arrow:xOrigin=312,yOrigin=164,xTarget=216,yTarget=86
Arrow:xOrigin=304,yOrigin=164,xTarget=216,yTarget=86
Arrow:xOrigin=304,yOrigin=166,xTarget=216,yTarget=86
Arrow:xOrigin=304,yOrigin=164,xTarget=208,yTarget=86
Arrow:xOrigin=304,yOrigin=164,xTarget=216,yTarget=84
Arrow:xOrigin=312,yOrigin=164,xTarget=216,yTarget=84
Arrow:xOrigin=312,yOrigin=164,xTarget=216,yTarget=84
Arrow:xOrigin=304,yOrigin=164,xTarget=216,yTarget=82
Arrow:xOrigin=312,yOrigin=164,xTarget=216,yTarget=84
Arrow:xOrigin=312,yOrigin=162,xTarget=216,yTarget=82 ← 线条矢量终点坐标
Arrow:xOrigin=312,yOrigin=162,xTarget=216,yTarget=82
Arrow:xOrigin=304,yOrigin=162,xTarget=216,yTarget=82
Arrow:xOrigin=312,yOrigin=162,xTarget=216,yTarget=80
Arrow:xOrigin=312,yOrigin=164,xTarget=216,yTarget=82
```

I2C模式

硬件连线图



HUSKYLENS常规设置

把HuskyLens的通讯模式设置为I2C。



也可以设置为“自动识别模式”，无需反复调节通讯协议，简单方便。

样例代码

```
#include "HUKSYLENS.h"
#include "SoftwareSerial.h"

HUKSYLENS huskylens;
//HUKSYLENS green Line >> SDA; blue Line >> SCL

void setup() {
    Serial.begin(115200);
    Wire.setClock(400000);
    Wire.begin(); // 初始化I2C连接
    while (!huskylens.begin(Wire)) //与huskylens通信失败
    {
        Serial.println(F("Begin failed!"));
        Serial.println(F("1.Please recheck the \"Protocol Type\" in HUKSYLENS (General Settings>>Protocol Type>>I2C)"));
        Serial.println(F("2.Please recheck the connection."));
        delay(500);
    }
}

void loop() {
    if (!huskylens.request()) Serial.println(F("Fail to request objects from HUKSYLENS!")); // 未从huskylens中请求对象信息
    else if(!huskylens.isLearned()) Serial.println(F("Object not learned!")); // huskylens处于待学习状态
    else if(!huskylens.available()) Serial.println(F("Object disappeared!")); // huskylens未识别到学习过的对象
    else
    {
        HUKSYLENSResult result = huskylens.read();
        if (result.command == COMMAND_RETURN_BLOCK)
        {

Serial.println(String()+F("Block:xCenter=")+result.xCenter+F(",yCenter=")+result.yCenter+
F(",width=")+result.width+F(",height=")+result.height); // 返回huskylens识别框的中心坐标和宽高
        }
        else if (result.command == COMMAND_RETURN_ARROW)
        {

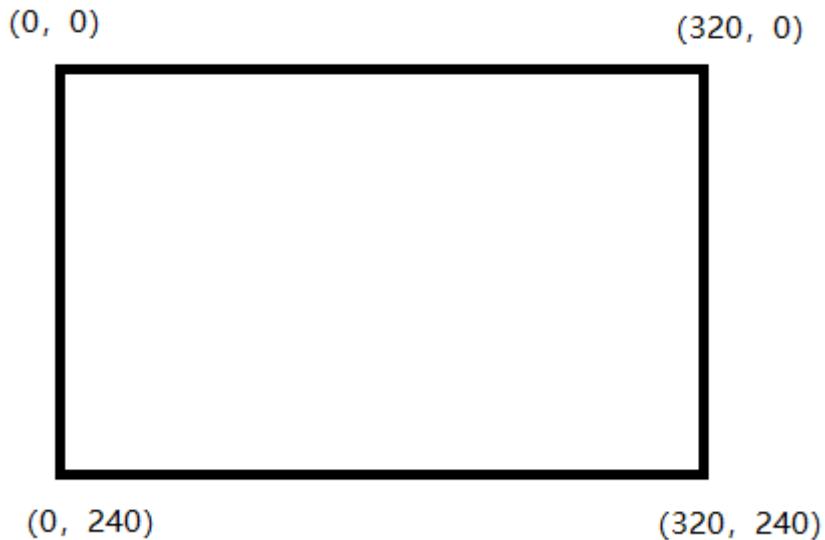
Serial.println(String()+F("Arrow:xOrigin=")+result.xOrigin+F(",yOrigin=")+result.yOrigin+
F(",xTarget=")+result.xTarget+F(",yTarget=")+result.yTarget); // 返回huskylens识别到的线条矢量的起点坐标与终点坐标
        }
        else
        {
            Serial.println("Object unknown!"); // huskylens检测到未学习过的对象
        }
    }
}
```

预期结果

1. 上传上述代码至Arduino主板
2. 参考前面讲解各个功能的章节，让HuskyLens学习完一个新东西，比如人脸。
3. 打开Arduino IDE的串口监视器查看结果。I2C模式下的结果与UART模式下的结果一样，此处不再赘述。

补充说明

HUSKYLENS处于巡线追踪模式时，其屏幕上的坐标系如下所示，这有助于你了解巡线模式下的点坐标值的由来。



Arduino案例2：履带小车巡线

更多案例，敬请期待。

更多文档

- [Arduino 库\(github\)](#)
- [micro:bit Makecode库\(github\)](#)
- [Mind+ 专题教程](#)
- [HuskyLens通信协议文档](#)
- [二维码标签样图](#)
- [色块样图](#)