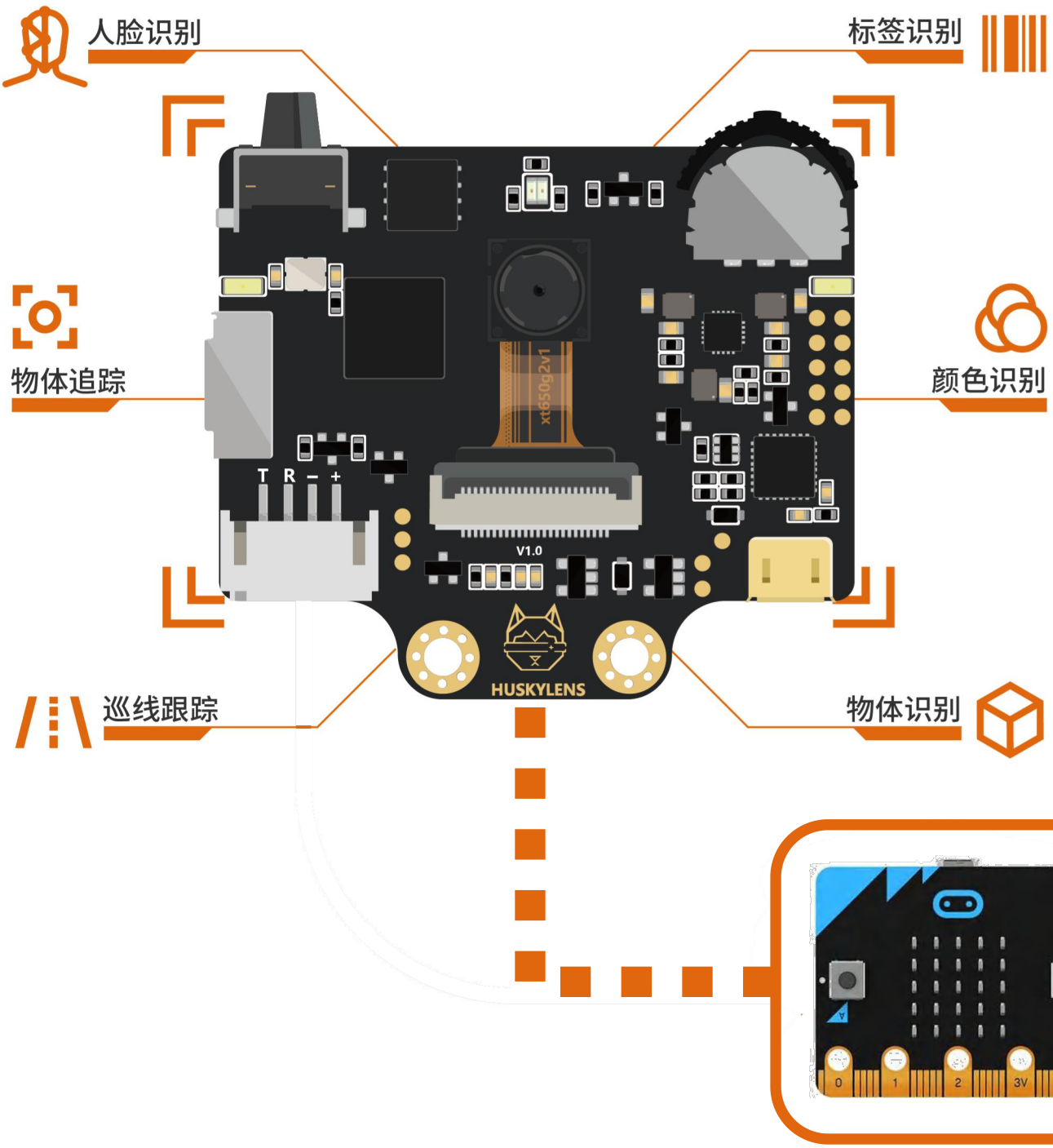
DFROBOT
DRIVE THE FUTURE

二哈助你慧眼识图

基于HUSKYLENS传感器项目教程



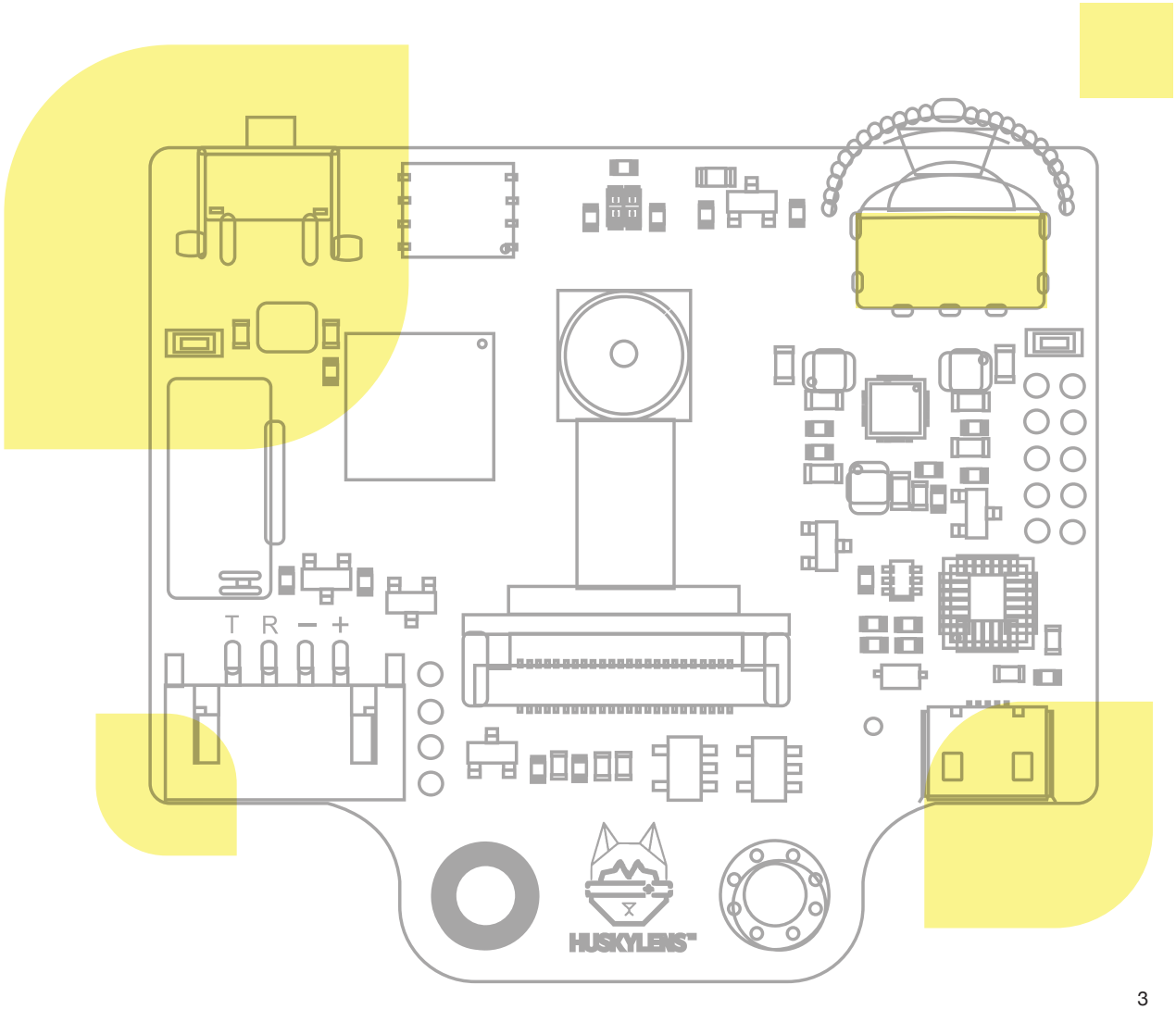
CONTENTS

目录

DFROBOT
DRIVE THE FUTURE

音乐魔镜

Project 1



魔镜魔镜，谁是这个世界上最美丽的人呢？大家看到这句话的时候，一定会想到小时候的童话故事《白雪公主和七个小矮人》，这是故事里面的恶毒王后和魔镜的经典台词。

现在有了HUSKYLENS就真的可以自己动手做这样一面魔镜喔，只不过让镜子说话还有点难度，但是让蜂鸣器响起来却很简单。所以就让我们一起来做一个音乐魔镜吧，让镜子播放不同的音乐来表达它的态度！

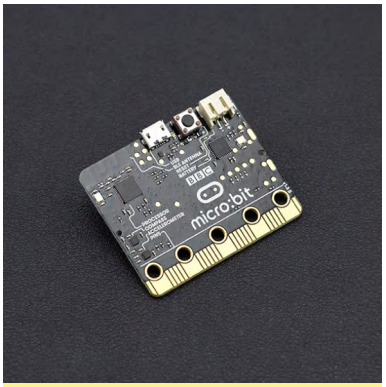
接下来一起看看如何制作吧！

功能介绍:

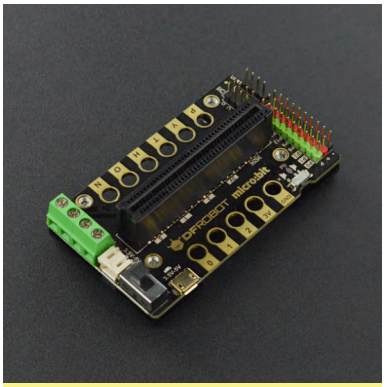
在这个项目中，我们将学习 HUSKYLENS 的人脸识别功能，利用其内置的机器学习技术,分辨学习过的人脸和其他人脸,并对应播放不同种类的音乐,这就是音乐魔镜。

想一想，当你把 HUSKYLENS 的摄像头对准你的脸，它会播放出表示赞美的美妙音乐乐，当你对准别人的脸，就会播放一些搞怪的音乐，会不会让大家一头雾水呢?

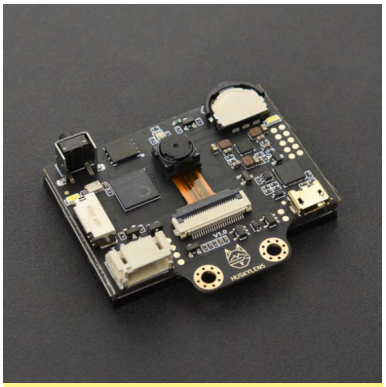
材料清单:



Micro:bit ×1



扩展板 ×1



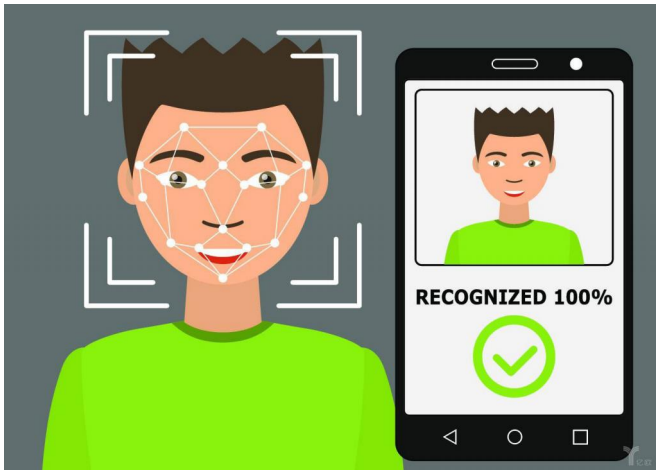
HUSKYLENS ×1

知识园地:

人脸与人体的其它生物特征（指纹、虹膜等）一样与生俱来，具有唯一性和不易被复制的特性。人脸图像属于最早被研究的一类图像，也是计算机视觉领域中应用最广泛的一类图像，这个项目就是利用了 HUSKYLENS 的**人脸识别**功能。

1. 什么是人脸识别?

人脸识别是基于**人的面部特征信息**进行身份识别的一种生物识别技术。使用摄像头或者摄像机采集含有人脸的图像或视频，自动检测图像信息和跟踪人脸，对检测到的人脸进行脸部的一系列相关分析技术。



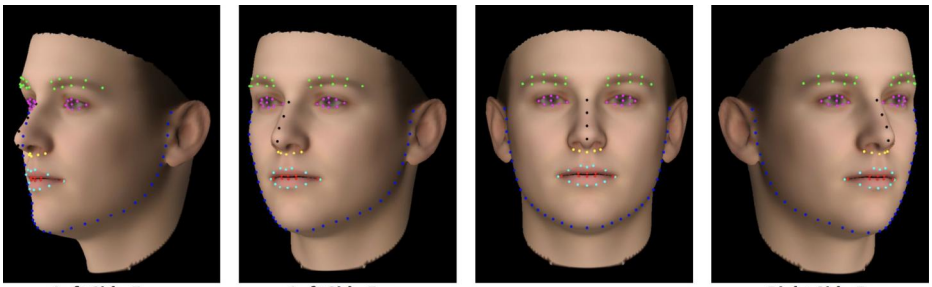
人脸识别工作原理:

人脸识别的过程中有 **4 个关键的步骤**:

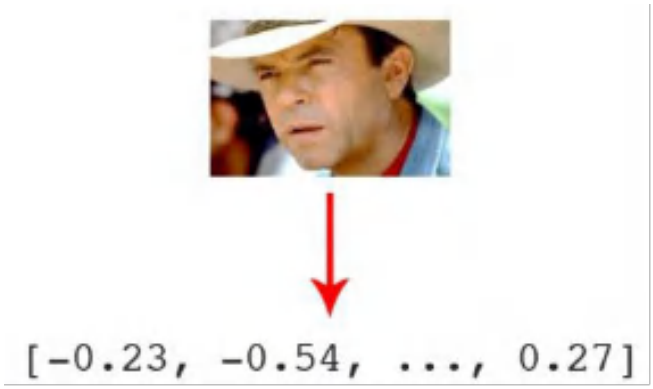
下面简单说明一下这 4 个步骤。

1 **人脸检测:** 寻找图片中人脸的位置，一般会用方框标出。

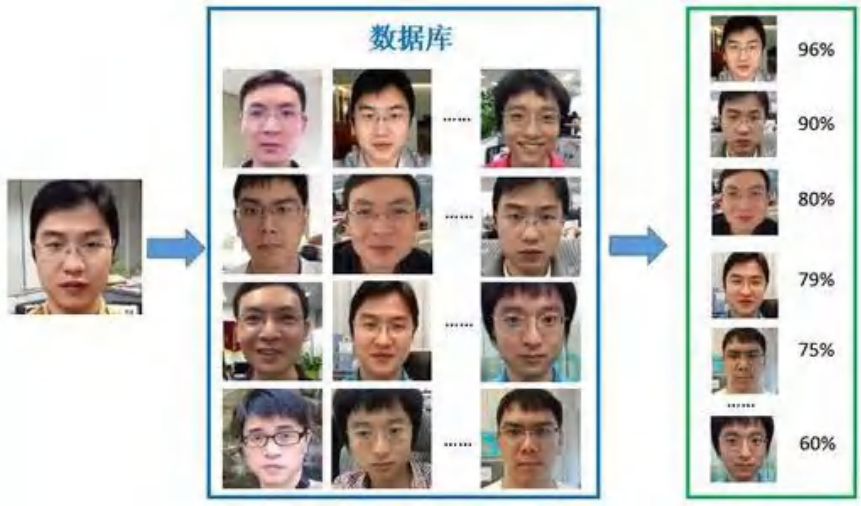
2 **人脸对齐：**通过定位人脸上的特征点，识别不同角度的人脸。



3 **人脸编码：**可以简单理解为提取人脸信息，转换为计算机可以理解的信息。



4 **人脸匹配：**将人脸信息匹配已有的数据库，从而得到一个相似度分数，给出匹配结果。



人脸识别也被认为是生物特征识别领域甚至人工智能领域最困难的研究课题之一。**人脸识别的困难**主要是人脸作为生物特征的特点所带来的。

相似性：不同个体之间的区别不大，所有的人脸的结构都相似，甚至人脸器官的结构外形都很相似。这样的特点对于利用人脸进行定位是有利的，但是对于利用人脸区分人类个体是不利的。

易变性：人脸的外形很不稳定，人可以通过脸部的变化产生很多表情，而在不同观察角度,人脸的视觉图像也相差很大,另外,人脸识别还受光照条件(例如白天和夜晚，室内和室外等）、人脸的很多遮盖物（例如口罩、墨镜、头发、胡须等）、年龄等多方面因素的影响。

人脸识别应用场景：

门禁系统：受安全保护的地区可以通过脸部辨识辨识试图进入者的身分，比如监狱、看守所、小区、学校等。

摄像监视系统：在例如银行、机场、体育场、商场、超级市场等公共场所对人群进行监视，以达到身分辨识的目的。例如在机场安装监视系统以防止恐怖分子登机。

网络应用：利用脸部辨识辅助信用卡网络支付，以防止非信用卡的拥有者使用信用卡，社保支付防止冒领等。

人脸识别目前在各行各业都有非常广泛的应用，例如学生考勤系统、相机、解锁手机、人证核验一体机等。

2. HUSKYLENS 人脸识别功能演示?

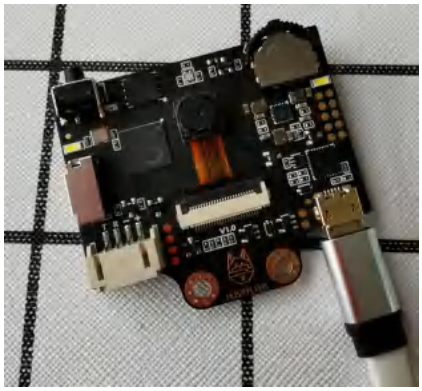
回到我们的音乐魔镜项目，HUSKYLENS 之所以能区分人脸就是因为其内置的机器学习功能，它就像一个数据库的采集者，可以手动录入指定的人脸信息，并且标记这个信息。

具体怎么操作呢? 先拿出你的 HUSKYLENS，让我们一起操作一遍。

第一次使用摄像头的用户请参照 WIKI 网址进行固件烧录和语言设置:
http://wiki2.dfrobot.com.cn/HUSKYLENS_V1.0_SKU_SEN0305_SEN0336

STEP1 连接电源

HUSKYLENS 自带独立 USB 供电口，连接 USB 线，即可开机。

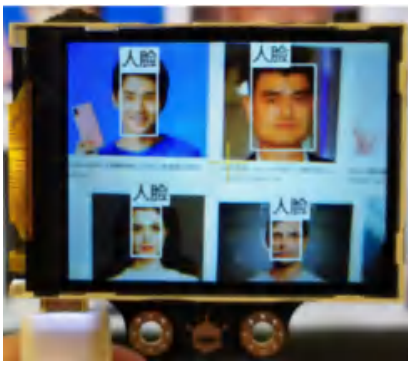


STEP2 选择“人脸识别”功能

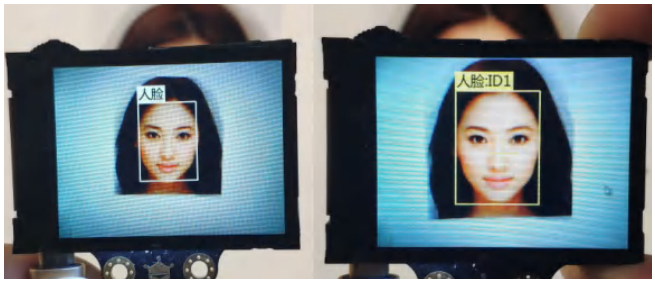
向左拨动“功能按键”，直至屏幕顶部显示“人脸识别”。

STEP3 学习人脸

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。



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。



- 长按 “学习按键” 不松开，可以多角度录入人脸。
- 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）。此时短按“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

项目实践：

学习完 HUSKYLENS 摄像头的基本操作后，让我们一起来完成音乐魔镜的制作吧～

首先要实现的功能就是摄像头在识别人脸时，能在程序端区分学习过的人脸和未学习过的。其次就是加入音乐，实现不同类音乐的播放，至少是两首。最后，可以找一面家中落灰多年的小镜子，进行外观搭建。那么我们分成两个任务来完成。

任务一：区分人脸

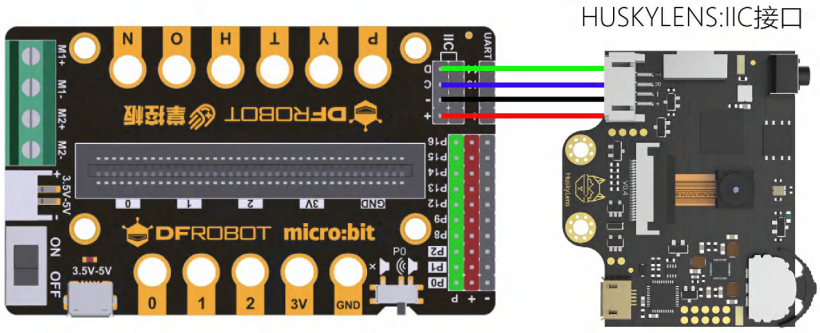
在这步我们先学习如何使用 HUSKYLENS 摄像头来识别并区分人脸，并能够判断是否是学习过的，并有一定反馈。

任务二：加入音乐

在学习了如何区分人脸并执行反馈功能后，我们就可以在此基础上添加更多的功能，比如在识别到不同人脸之后能够播放不同的音乐。

任务一：区分人脸

1 硬件连接



2 程序设计

这里的区分人脸默认摄像头已经学习过指定人脸信息当摄像头再次看到人脸时，判断是否是学习过的即可。为了让显示过程更加直白，当主控板屏幕上显示笑脸表示识别到的是指定人脸，显示哭脸表示不是。

在正式编程前，通过以下 3 步先进行程序设计。

STEP1 Mind+ 软件设置

打开 Mind+ 软件（1.62 或以上版本），切换到“上传模式”，点击“扩展”，在“主控板”下点击加载 “micro:bit”，在 “传感器” 下点击加载 “HUSKYLENS AI 摄像头”。

STEP2 指令学习

来认识一下主要用到的几条指令。

初始化，仅需执行一次，放在主程序开始和循环执行之间，可选择 I2C 或串口,I2C 地址不用变动。**注意 HUSKYLENS 端需要在设置中调整“输出协议”与程序中一致，否则读不出数据。**

切换算法，可以随时切换到其他算法，同时只能存在一个算法，注意切换算法需要一些时间。

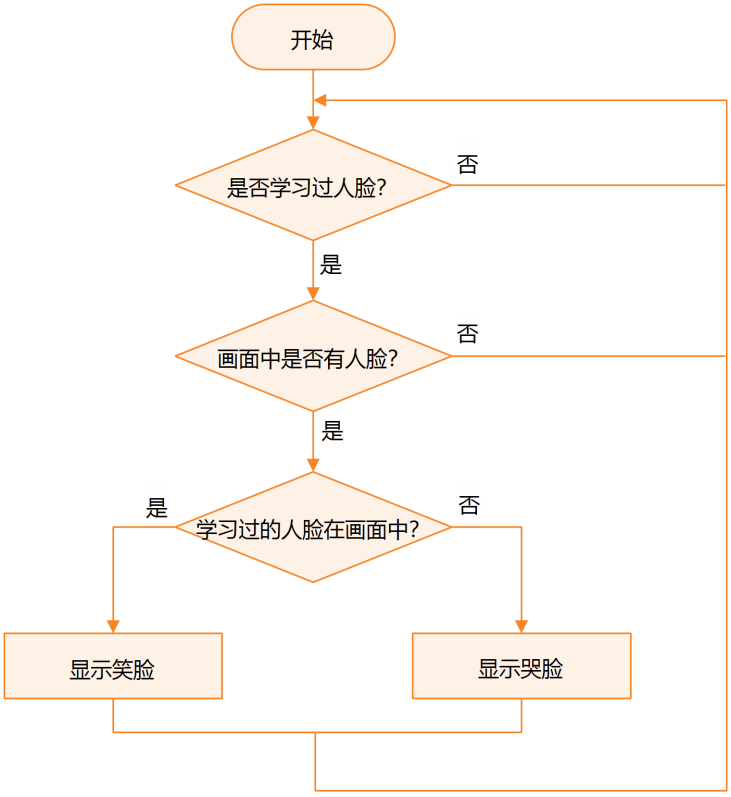
主控板向 HUSKYLENS 请求一次数据存入“结果”（存在主控板的内存变量中，一次请求刷新一次存在内存中的数据），之后可以从“结果”中获取数据，此模块调用之后“结果”中才会获取到最新的数据。

从请求得到的“结果”中获取当前界面中是否有方框或箭头，包含已学习(id 大于 0)和未学习的，有一个及以上则返回 1。

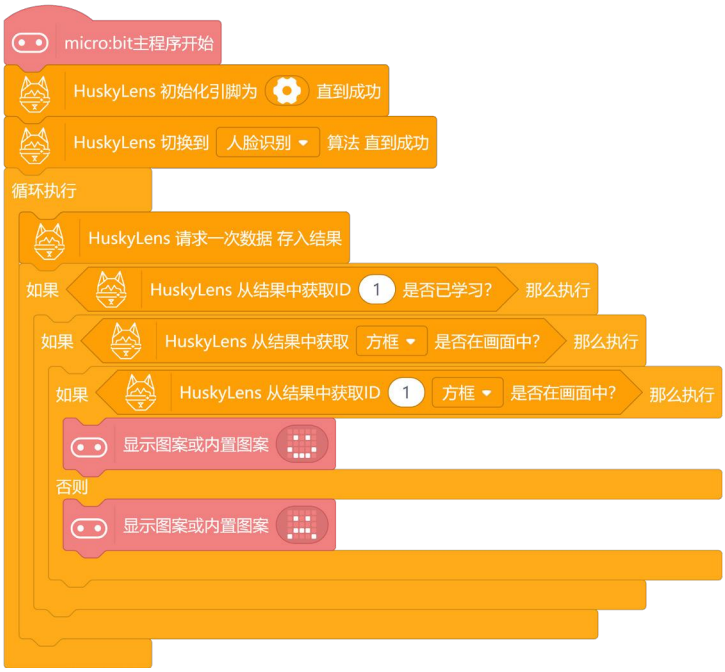
从请求得到的“结果”中获取是否 IDx 已经进行了学习

从请求得到的“结果”中获取是否 IDx 在画面中，方框指屏幕上目标为方框的的算法，箭头对应屏幕上目标为箭头的算法，当前仅为巡线算法时选择箭头，其他算法都选择方框。

STEP3 流程图分析

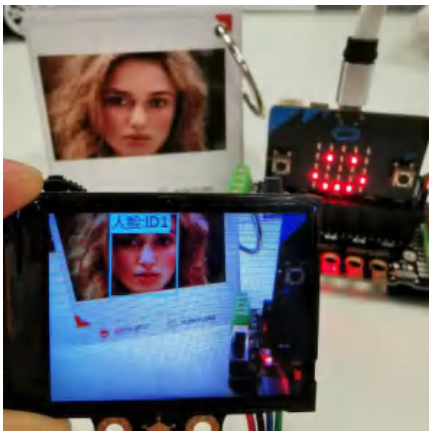


3 程序示例



4 运行效果

摄像头提前学习好一张人脸,将上面程序上传到主控板后当摄像头看到指定人脸,则显示笑脸表情,当看到其他人脸,就会切换为哭脸表情。



运行程序时,除了给主控板供电,摄像头也需要单独供电

任务二：加入音乐

1 程序设计

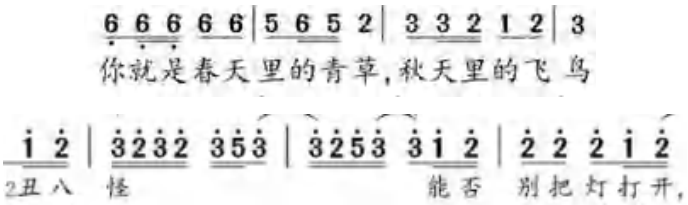
在这里，我们可以在笑脸和哭脸表情下各加入一段音乐，一首赞美的，一首搞怪的。比如时下很火的《百花香》里的“你就是春天里的青草，秋天里的飞鸟”和《丑八怪》里的“丑八怪，能否别把灯打开”。

如何加入音乐呢？

这里使用的扩展板自带蜂鸣器，使用时将蜂鸣器开关打开即可。

如何找乐谱呢？

上网搜索《百花香》、《丑八怪》简谱，截取需要的段落。



如何编写程序呢？

在 Mind+，自带播放音符指令，分为低、中、高音，还有各种节拍。

如何将乐谱与指令对应呢？

这里提供一个简单的识别方法，以音符 2 为例，如下表。

低音	2	1 拍	2
中音	2	1/2 拍	2
高音	2	1/4 拍	2

2 程序示例

3 运行效果

提前在摄像头上学好你自己的人脸，将程序上传到主控板中，运行程序时摄像头只要看见你,就会唱一段“你就是春天里的青草,秋天里的飞鸟”,如果摄像头对着其他人，就会唱丑八怪，能否别把灯打开”，怎么样，是不是会把别人搞懵呀。

最后，可以找一面的小镜子，把硬件藏在镜子背面，露出摄像头采集人脸信息，搭建出一面音乐魔镜。

项目小结:

通过音乐魔镜项目，我们了解了人脸识别的工作原理，学习了 HUSKYLENS 上人脸识别算法的指令应用。

在人工智能视觉识别领域,人脸识别是不可或缺的一部分,也有丰富的应用场景，大家一起来开动脑筋，想一想还能做出哪些人脸识别的应用呢。

知识点回顾:

1、了解人脸识别的工作原理

2、学习 HUSKYLENS 人脸识别的相关指令

项目拓展：

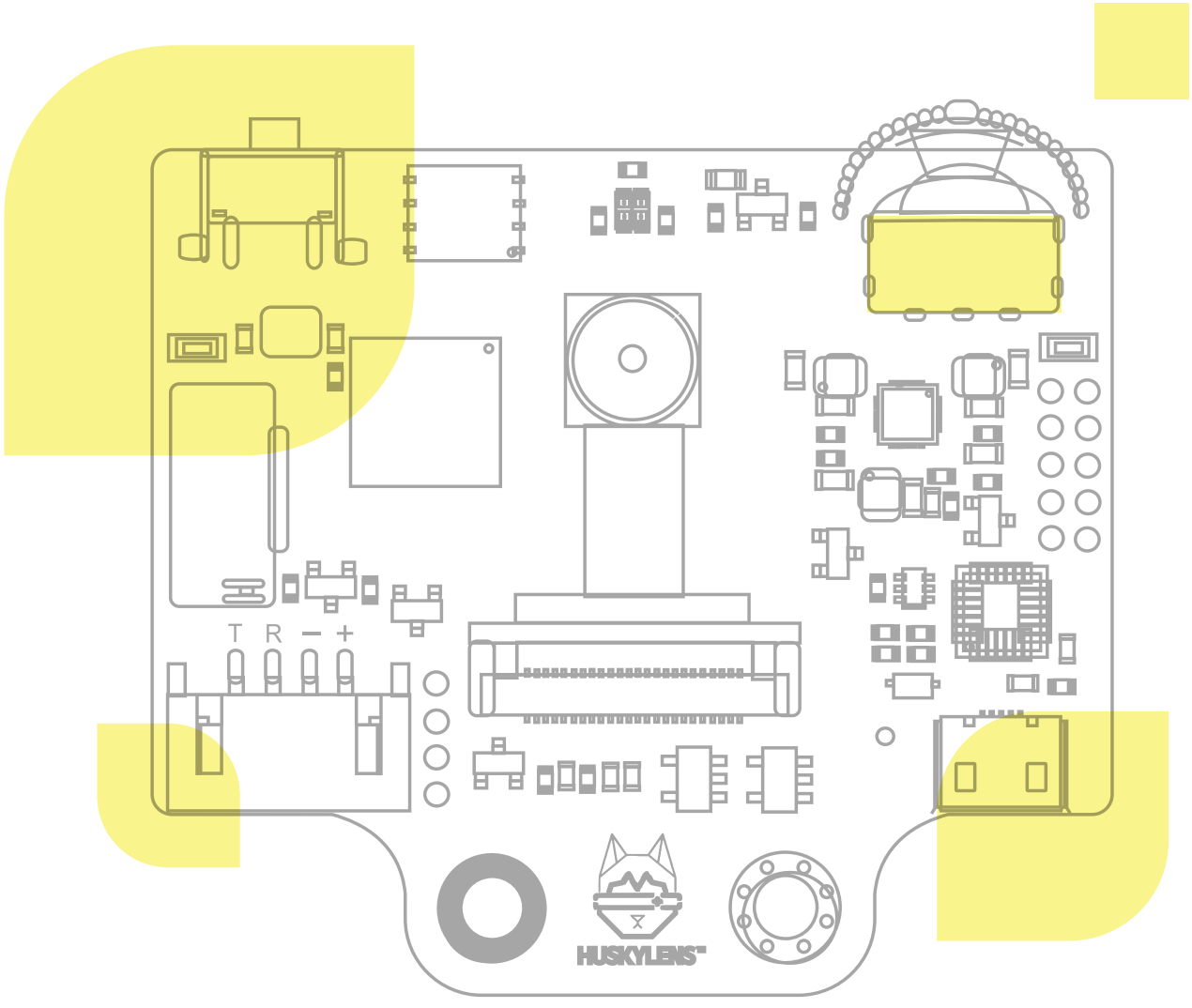
这个项目中，我们将人脸识别与音乐结合，做了一个好玩的音乐魔镜。

其实回头想一想，摄像头识别到不认识的人脸，就发出指定声音，这不就是门禁警报系统嘛！但是如果应用在实际场景中，蜂鸣器报警的声音实在太小，所以能不能将它与物联网相结合呢？如果你的手边刚好有物联网模块，尝试做一个家庭警报系统吧，利用摄像头检测门口是否有陌生人，如果陌生人停留的时间过长，那么利用物联网发送消息到主人的电脑或者手机上。

DFROBOT
DRIVE THE FUTURE

色彩钢琴

Project 2



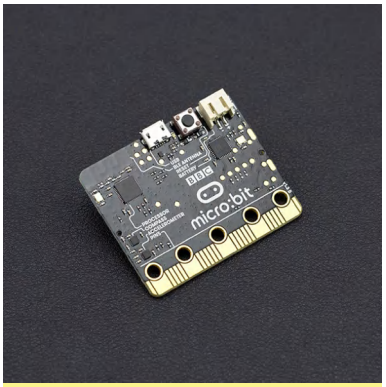
你是否有过一个音乐梦呢，是否想象过能想一名钢琴家一样优雅的弹琴。“乐器之王”钢琴以行云流水般的音符阐释着完美的音效和浪漫情怀，为人们带来纯净的享受。但是由于种种原因也许你没有学过钢琴或是没能拥有一台钢琴。

现在有了 HUSKYLENS 传感器，我们也可以亲手为自己制作一台色彩钢琴，实现你的音乐梦，让我们利用彩色的琴键发出美妙的音乐吧。

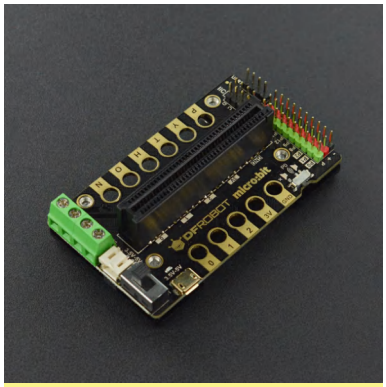
功能介绍:

本项目利用 HuskyLens 的颜色识别功能，通过识别不同颜色的琴键，播放不同的音符，让你的 “演奏” 既好看又好听，拥有绝对美妙的舞台效果。

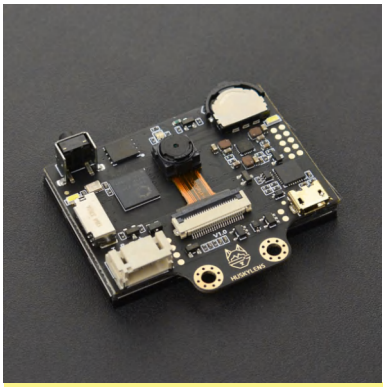
材料清单:



Micro:bit ×1



扩展板 ×1



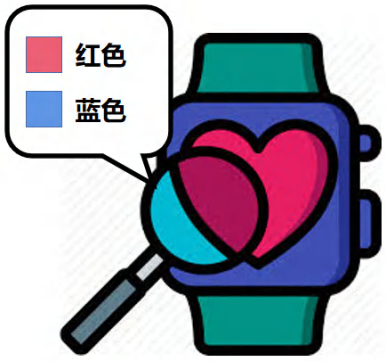
HUSKYLENS ×1

知识园地:

当今社会,自动化生产已经成为了社会的发展趋势,机器视觉作为“机器人”的眼睛，则显得尤为重要.颜色识别作为其中一个重要的技术方向,已经经历了多代技术的升级。而我们这个项目就是借助 HUSKYLENS 传感器的颜色识别功能来对色彩进行区分和识别，通过不同的颜色奏响我们的小钢琴。

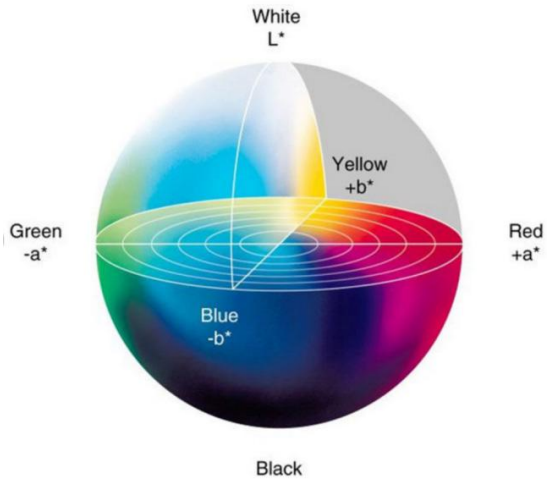
1. 什么是颜色识别?

什么是颜色识别，首先我们要了解知道什么是颜色，颜色是通过眼、脑和我们的生活经验所产生的一种对光的视觉效应，我们肉眼所见到的光线，是由波长范围很窄的电磁波产生的，不同波长的电磁波表现为不同的颜色。颜色识别就是基于不同亮度下的色彩属性进行识别和区分的。



颜色识别工作原理:

颜色识别是基于 **Lab 色彩空间** (英语: Lab color space) 进行识别的，带有维度 L* 代表亮度，a* 代表从绿色到红色的分量 ,b* 代表从蓝色到黄色的分量，基于非线性压缩的 CIE XYZ 色彩空间坐标。我们可以将 Lab 这三个参数理解为三维坐标系的 XYZ。对已经识别学习的颜色的 Lab 参数进行比对，当两个颜色在一定的误差范围内相吻合时，就判定为是同一个颜色。

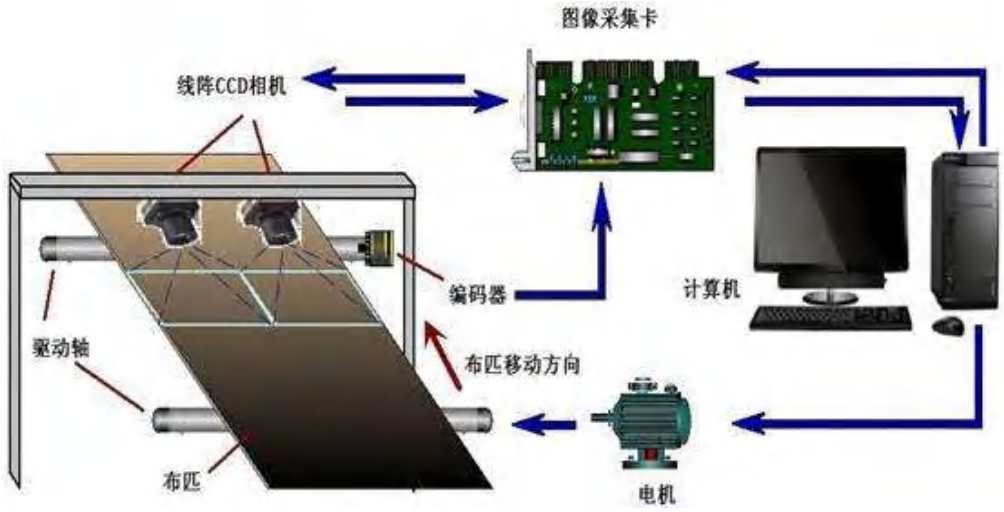


Lab 坐标系

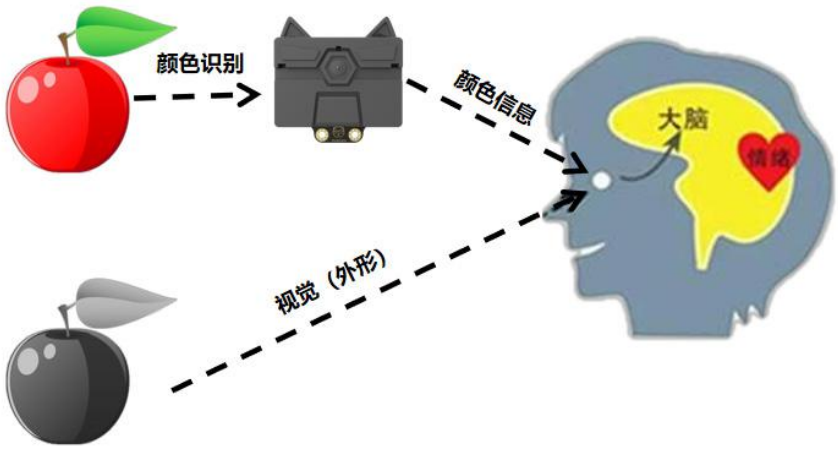
在我们平时使用颜色识别中，同一个模块的颜色属性中色相和饱和度是固定不变的，但是受到环境亮度的影响明亮度会产生一些变化，所以在使用颜色识别功能时一定要保证学习识别时的环境亮度和实际工作时的环境亮度尽量保持一致。

颜色识别主要应用在以下领域：

1. 在工业领域使用，颜色识别目前在工业领域使用较多，如印刷、涂料和纺织品等行业，用于色彩监视和校准等工作。



2. 对色弱或视觉障碍的人进行辅助识别，能增强他们对颜色和色彩的理解。



2.HUSKYLENS 传感器 - 颜色识别功能演示?

如果想要色彩钢琴能够顺利的演奏起来，首先要让 HUSKYLENS 传感器学习我们的彩色琴键的颜色，并让它知道每个颜色对应的音符。HUSKYLENS 传感器中的颜色识别功能是利用传感器内置算法，通过对不同颜色进行学习和记录，能够辨别出不同颜色的 ID 并反馈给主控板。

在 HuskyLens 传感器中默认设置为只学习、识别并追踪一种颜色，但是我们的彩色琴键肯定不能只有一个，所以我们需要将其设置为能够识别多种颜色的状态。

操作设置 - 学习多个:

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。

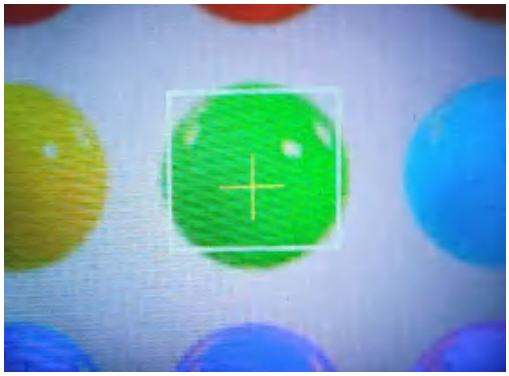
4. 向左拨动 “功能按键”，选中 “保存并返回 “，短按” 功能按键 “，屏幕提示” 是否保存参数？“，默认选择” 确认 “，此时短按” 功能按键 “，即可保存参数，并自动返回到颜色识别模式。

这样我们就设置好学习多个的功能了。

学习与识别

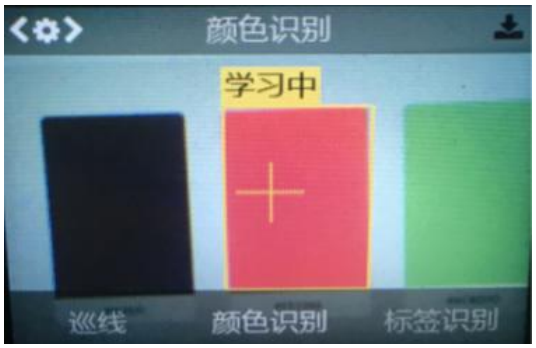
1. 侦测颜色

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

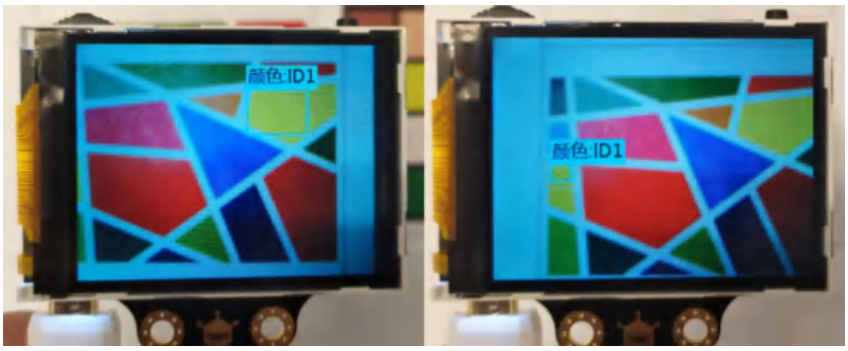
侦测到颜色后，按下 “学习按键” 学习第一种颜色，然后松开 “学习按键” 结束学习，屏幕上有消息提示：“再按一次继续，按其他按键结束”。如要继续学习下一种颜色，则在倒计时结束前按下 “学习按键”，可以继续学习下一种颜色。如果不再需要学习其他颜色了，则在倒计时结束前按下” 功能按键” 即可，或者不操作任何按键，等待倒计时结束。HuskyLens 显示的颜色 ID 与学习颜色的先后顺序是一致的，也就是：ID 会按顺序依次标注为 “ID1”，“ID2”，“ID3”，以此类推，并且不同颜色对应的边框颜色也不同。



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。

项目实践：

我们将分为两步将任务完成，首先我们会学习使用 HuskyLens 的颜色识别功能，并将识别到的颜色 ID 输出。然后我们就可以根据输出的颜色 ID 给他们对应的声音播放，这样就可以完成我们的色彩钢琴了。我们将通过两个任务来完成色彩钢琴。

任务一：多种颜色识别

在最开始我们需要让 HuskyLens 摄像头能够识别多种颜色，并区分这些颜色的不同，能够给出反馈，以便于后续我们增加音符。

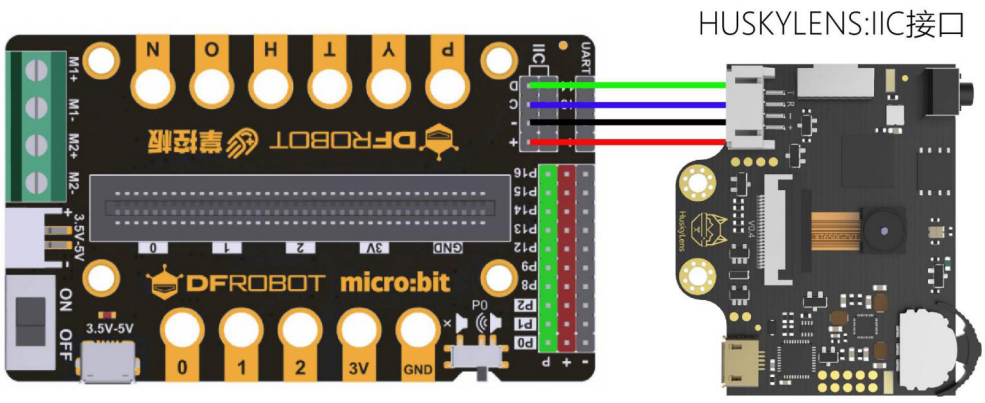
任务二：给每个颜色定义音符

在能够精准的识别出每个颜色之后，我们就可以给每个颜色定义一个声音，让他们能够按照一定的规律播放，这样就可以实现色彩钢琴了。

任务一：多种颜色识别

1 硬件连接

HuskyLens 传感器使用的是 IIC 接口，需要注意线序，不要接错或接反。

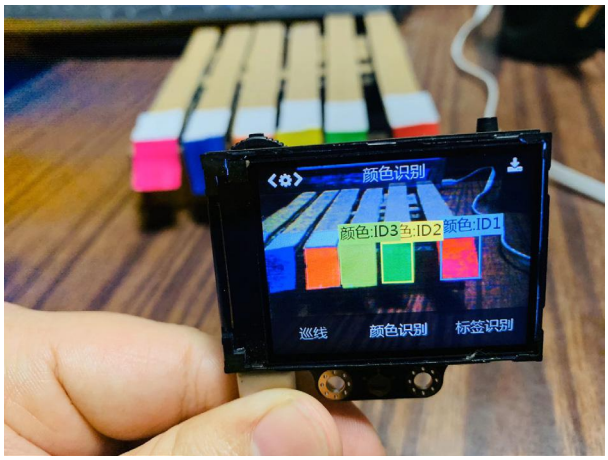


2 程序设计

这里我们需要让 HuskyLens 传感器学习各个琴键的颜色，并能够输出颜色 ID 以方便后续我们对应颜色奏响对应的声音

STEP1 学习与识别

在设计程序之前我们需要让 HuskyLens 传感器学习各个琴键的颜色。（注意需要先开启学习多个的功能）



STEP2 Mind+ 软件设置

打开 Mind+ 软件（1.62 或以上版本），切换到“上传模式”，点击“扩展”，在“主控板”下点击加载 “micro:bit”，在 “传感器” 下点击加载 “HUSKYLENS AI 摄像头”。

STEP3 指令学习

来认识一下主要用到的几条指令。

初始化，仅需执行一次，放在主程序开始和循环执行之间，可选择 I2C 或串口,I2C 地址不用变动。**注意 HUSKYLENS 端需要在设置中调整“输出协议”与程序中一致，否则读不出数据。**

切换算法，可以随时切换到其他算法，同时只能存在一个算法，注意切换算法需要一些时间。

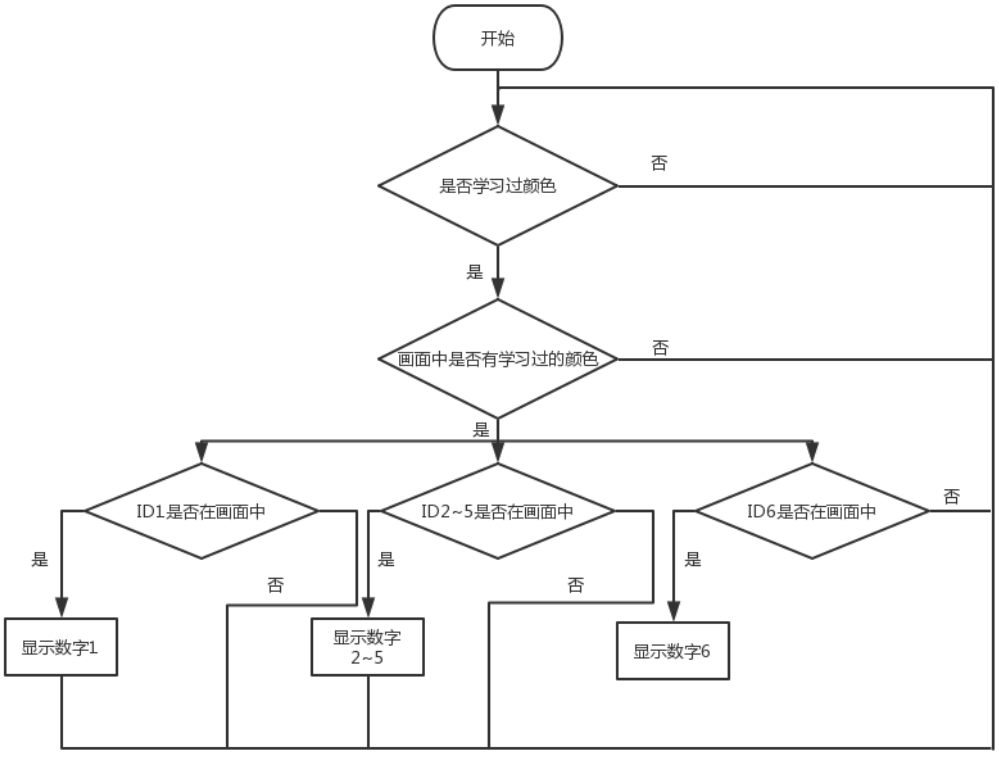
主控板向 HUSKYLENS 请求一次数据存入“结果”（存在主控板的内存变量中，一次请求刷新一次存在内存中的数据），之后可以从“结果”中获取数据，此模块调用之后“结果”中才会获取到最新的数据。

从请求得到的“结果”中获取当前界面中是否有方框或箭头，包含已学习(id 大于 0)和未学习的，有一个及以上则返回 1。

从请求得到的“结果”中获取是否 IDx 已经进行了学习

从请求得到的“结果”中获取是否 IDx 在画面中，方框指屏幕上目标为方框的的算法，箭头对应屏幕上目标为箭头的算法，当前仅为巡线算法时选择箭头，其他算法都选择方框。

STEP4 流程图分析

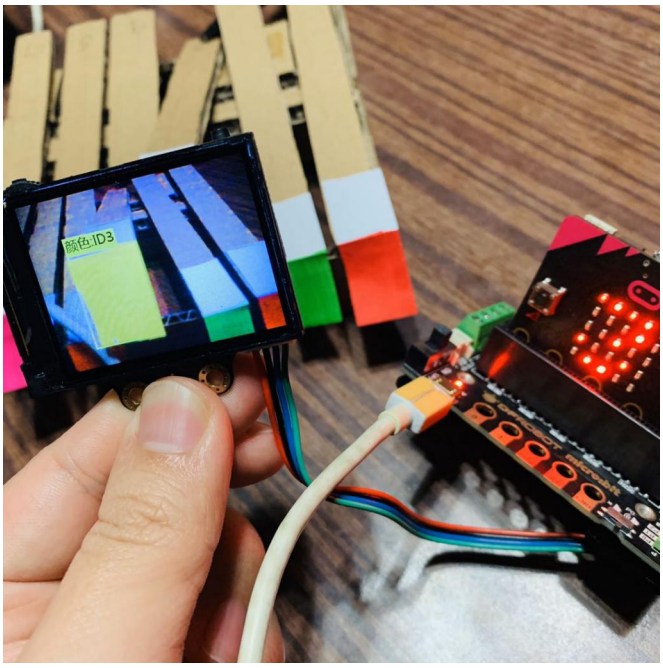


3 程序示例


micro:bit主程序开始
HuskyLens 初始化引脚为 直到成功
HuskyLens 切换到 颜色识别 算法 直到成功
循环执行
HuskyLens 请求一次数据 存入结果
如果 HuskyLens 从结果中获取ID 1 是否已学习? 那么执行
如果 HuskyLens 从结果中获取 方框 是否在画面中? 那么执行
如果 HuskyLens 从结果中获取ID 1 方框 是否在画面中? 那么执行
显示文字 1
接口 P0 播放音符 1 中 C/C4 1 拍
等待 0.1 秒
如果 HuskyLens 从结果中获取ID 2 方框 是否在画面中? 那么执行
显示文字 2
接口 P0 播放音符 2 中 D/D4 1 拍
等待 0.1 秒
如果 HuskyLens 从结果中获取ID 3 方框 是否在画面中? 那么执行
显示文字 3
接口 P0 播放音符 3 中 E/E4 1 拍
等待 0.1 秒
如果 HuskyLens 从结果中获取ID 4 方框 是否在画面中? 那么执行
显示文字 4
接口 P0 播放音符 4 中 F/F4 1 拍
等待 0.1 秒
如果 HuskyLens 从结果中获取ID 5 方框 是否在画面中? 那么执行
显示文字 5
接口 P0 播放音符 5 中 G/G4 1 拍
等待 0.1 秒
如果 HuskyLens 从结果中获取ID 6 方框 是否在画面中? 那么执行
显示文字 6
接口 P0 播放音符 6 中 A/A4 1 拍
等待 0.1 秒
如果 HuskyLens 从结果中获取ID 7 方框 是否在画面中? 那么执行
显示文字 7
接口 P0 播放音符 7 中 B/B4 1 拍
等待 0.1 秒

4 程序示例

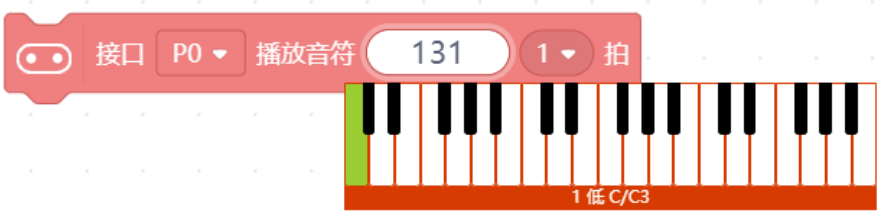
当在 HUSKYLENS 传感器中识别到对应颜色的 ID，就在 micro:bit 上显示对应的数字。



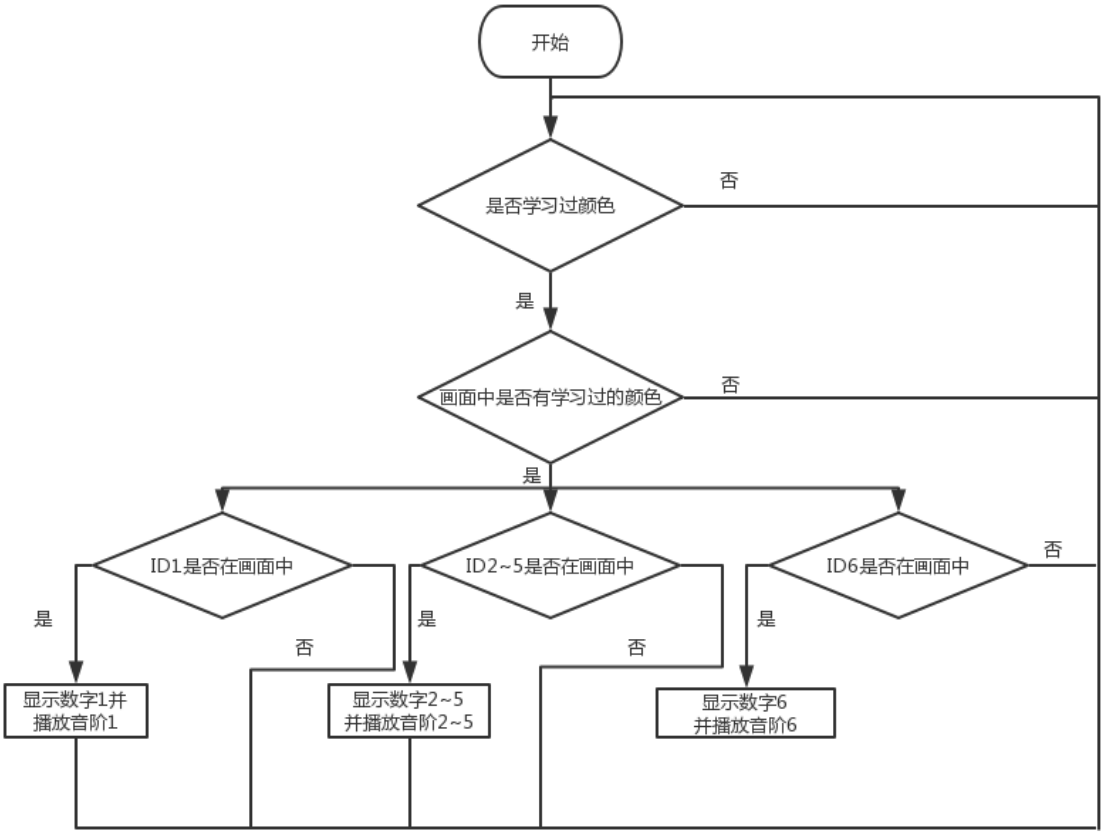
任务二：给每个颜色定义音符

1 硬件连接

如何编写程序呢？ 在 Mind+，自带播放音符指令，分为低、中、高音，还有各种节拍。我们只要给对应的 ID 添加对应的音符即可。



程序逻辑图如下：



2 程序示例

在上步完成的程序中添加音符模块即可。


micro:bit主程序开始
HuskyLens 初始化引脚为 直到成功
HuskyLens 切换到 颜色识别 算法 直到成功
循环执行
HuskyLens 请求一次数据 存入结果
如果 HuskyLens 从结果中获取ID 1 是否已学习? 那么执行
如果 HuskyLens 从结果中获取 方框 是否在画面中? 那么执行
如果 HuskyLens 从结果中获取ID 1 方框 是否在画面中? 那么执行
显示文字 1
等待 0.1 秒
如果 HuskyLens 从结果中获取ID 2 方框 是否在画面中? 那么执行
显示文字 2
等待 0.1 秒
如果 HuskyLens 从结果中获取ID 3 方框 是否在画面中? 那么执行
显示文字 3
等待 0.1 秒
如果 HuskyLens 从结果中获取ID 4 方框 是否在画面中? 那么执行
显示文字 4
等待 0.1 秒
如果 HuskyLens 从结果中获取ID 5 方框 是否在画面中? 那么执行
显示文字 5
等待 0.1 秒
如果 HuskyLens 从结果中获取ID 6 方框 是否在画面中? 那么执行
显示文字 6
等待 0.1 秒
如果 HuskyLens 从结果中获取ID 7 方框 是否在画面中? 那么执行
显示文字 7
等待 0.1 秒

3 运行效果

将 HuskyLens 传感器固定好，固定的位置在没按下琴键时，彩色琴键要在识别范围外，而摁下琴键时出现在识别范围内。当我们摁下琴键，会根据识别到的颜色播放对应的音符。

项目小结：

项目回顾：

本节课学习了解了颜色识别的工作原理，并通过使用 HuskyLens 传感器学习了传感器的颜色识别功能。

颜色识别在人工智能视觉识别中非常重要的一个功能，在工业中有着广泛的引用。大家想想颜色识别还可以实现什么有趣的功能？

知识点回顾：

1、了解颜色识别的工作原理；

2、学习了 HuskyLens 传感器的颜色识别功能和识别多个的操作方法。

项目拓展：

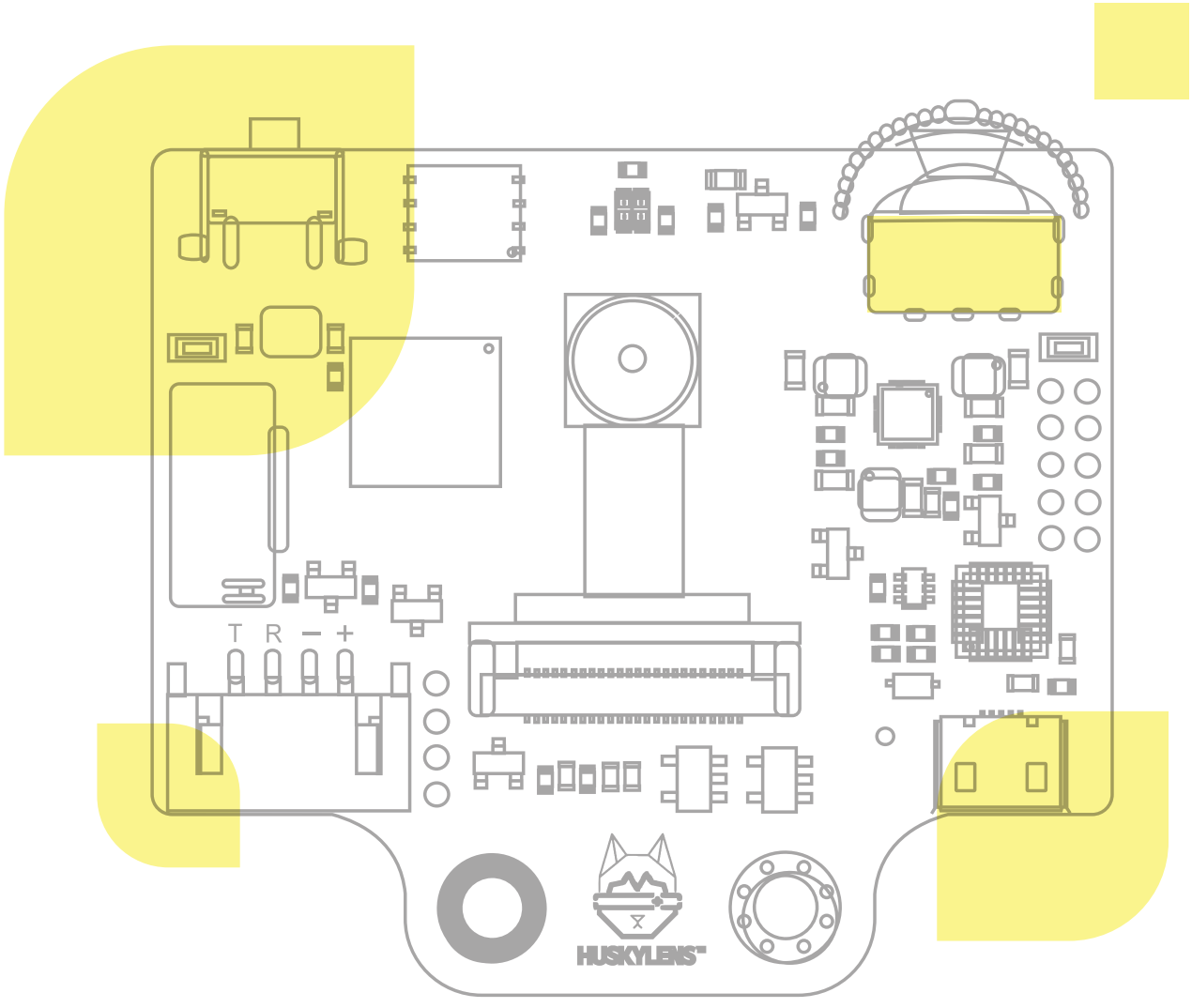
完成了彩色小钢琴之后，我们一定会发现一个问题，琴键的数量比较少，而如果我们要增加琴键的数量，随着颜色的增多，会有很多颜色相近的琴键，就有可能会出现误识别。琴键变多也会出现摄像头识别范围无法读取足够多的琴键。所以我们有没有什么办法扩宽我们小钢琴的音域呢？

> **小提示：**可以利用 micro:bit 上的 AB 按键实现升阶和降阶的功能。

DFROBOT
DRIVE THE FUTURE

智能的追光灯

Project 3



我们在电视上或者剧院中观看节目时，我们经常会看到一道追光灯打到演员身上,并随着演员的动作而移动,是不是非常耀眼醒目?

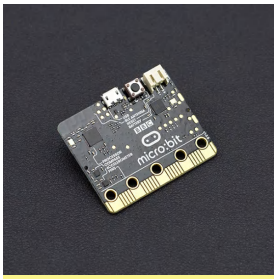
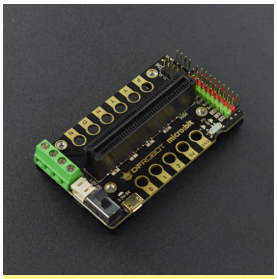
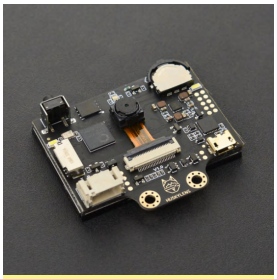
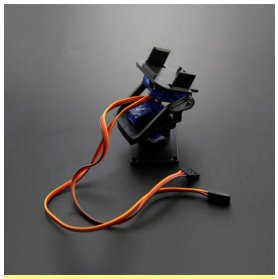
你知道追光灯在舞台上都是怎么实现追光的吗? 追光灯在实际应用中往往还是使用人工或者编程控制的方式进行，人工的方式需要追光师与演员多次彩排配合才能够实现出完美的舞台效果，而编程控制则是提前编写入固定的路径，使追光灯按照设置好的路径移动，这样对演员的限制又很多，如果稍微走偏几步就会影响舞台效果。

所以如果能有一个智能自动的追光灯，能自动识别演员去进行追光相信是对舞台表演会有很大的帮助。如果想要做舞台上最“亮”的仔，那自动追光灯是必不可少的，但是如何让追光灯能够跟上你的步伐，就需要一个操作精准的追光灯 “师傅” 了

功能介绍:

本项目利用 HuskyLens 的物体追踪功能，通过学习舞台上的人，实现对其进行追踪，驱动云台自动追踪，从而驱动灯光实时追踪你的脚步。

材料清单:

			
Micro:bit ×1	扩展板 ×1	HUSKYLENS ×1	迷你 2 自由度云台 ×1 DF9GMS 180° 微型舵机 x2

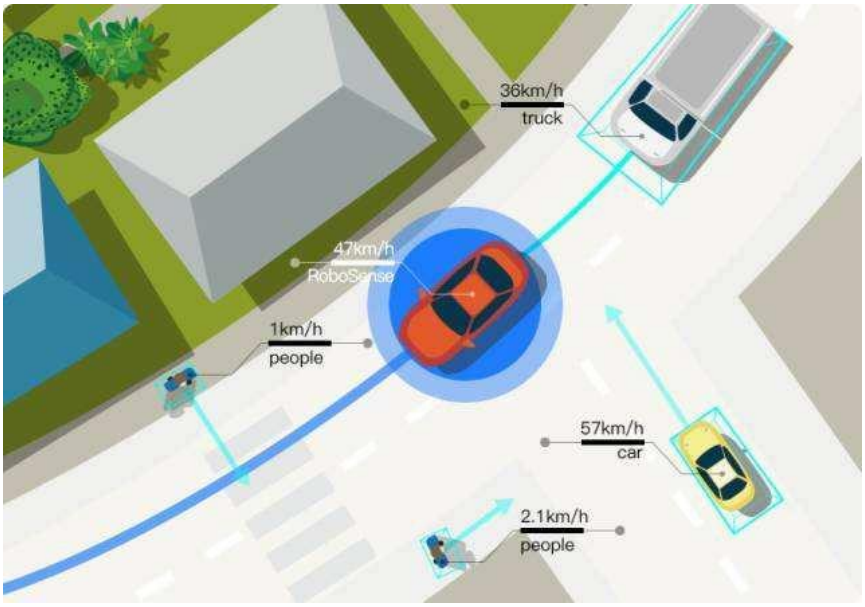
知识园地:

当我们需要追踪一个活动的物体，除了人工操作以外就需要使用到视觉物体追踪技术了，这项技术广泛应用在我们的生活中，如视频监控、无人机跟随拍摄等。我们这个项目就是利用 HUSKYLENS 的**物体追踪**功能来实现智能追光跟随的。

一、什么是物体追踪?

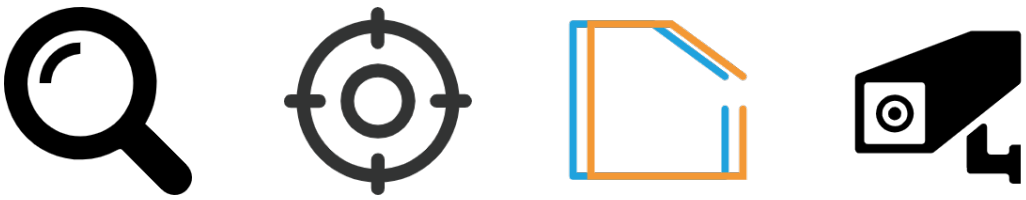
物体追踪是人工智能视觉识别中非常重要的一个功能，属于物体行为识别的一个类别。物体追踪是计算机视觉中的一项重要任务，是指对视频序列中的目标状态进

行持续推断的过程，简单来说就是识别指定目标并追踪或者追踪摄像头视觉范围内移动的物体。物体追踪技术在军事和民用方面都有着十分广泛的应用。



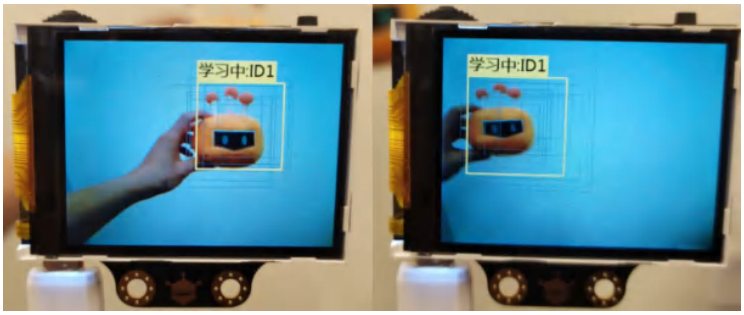
物体追踪工作原理：

通过单摄像头采集图像，将图像信息传入计算机，经过分析处理，计算出运动物体的相对位置，同时控制摄像头转动，对物体进行实时追踪。物体追踪系统执行追踪功能时主要分为四步，识别物体、追踪物体、预测物体运动、控制摄像头。

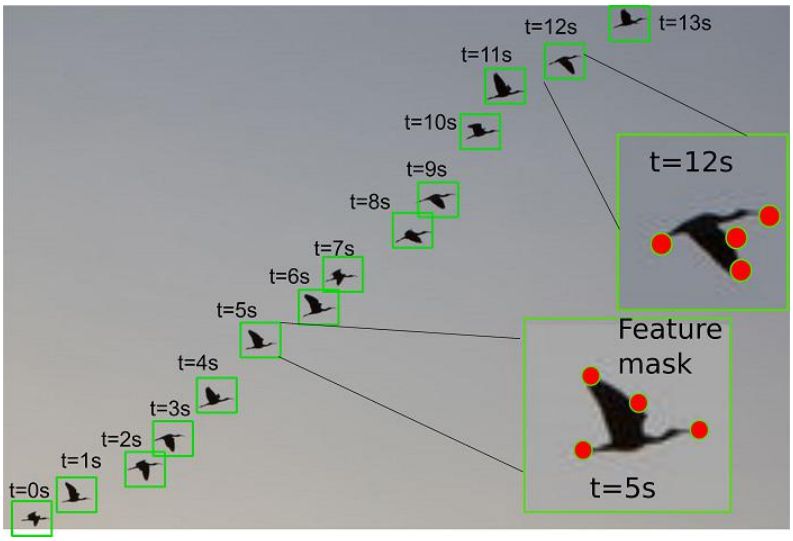


识别物体是在静态背景下，通过一些图像处理的算法，得到较为精确的物体的外观信息，能够识别出物体的形态并标注出来，如图所示。

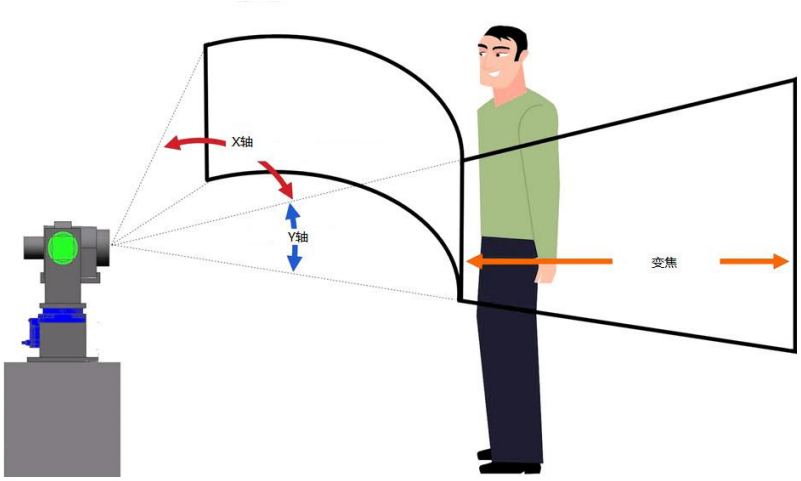
追踪物体是通过上一步得到的物体的外观信息，根据物体的外观特征，使用算法对后面的图像序列进行跟踪，并且可以在后续的追踪中进行更深入的学习，使追踪越来越精确。如图中，在移动中缓慢旋转物体，可以对其进行多角度学习。



预测物体运动是为了提高效率采用算法进行计算，预测下一帧运动物体图像的位置。可以优化算法并提高效率。如图可以通过前几秒钟鸟的移动趋势来预测后续的移动路径和动作。



控制摄像头在采集图像信息的同时移动摄像头，使摄像头跟随着物体移动的方向调整方向，一般需要配合云台或其他运动机构来实现。



物体追踪技术主要应用在以下领域：

1. 智能视频监控：

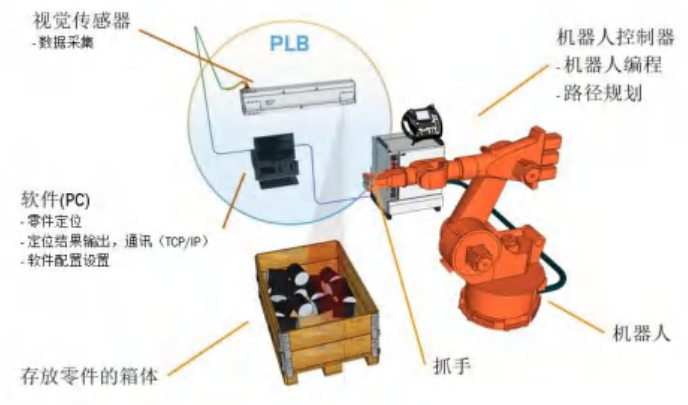
基于运动识别（基于步法的人类识别、自动物体检测等），自动化监测（监视一个场景以检测可疑行为）；交通监视（实时收集交通数据用来指挥交通流动）

2. 人机交互：

传统人机交互是通过计算机键盘和鼠标进行的，为了使计算机具有识别和理解人的姿态、动作、手势等能力，跟踪技术是关键

3. 机器人视觉导航:

在智能机器人中，跟踪技术可用于计算拍摄物体的运动轨迹



4. 虚拟现实:

虚拟环境中 3D 交互和虚拟角色动作模拟直接得益于视频人体运动分析的研究成果，可给参与者更加丰富的交互形式，人体跟踪分析是其关键技术

二、HUSKYLENS 传感器 - 物体追踪功能演示

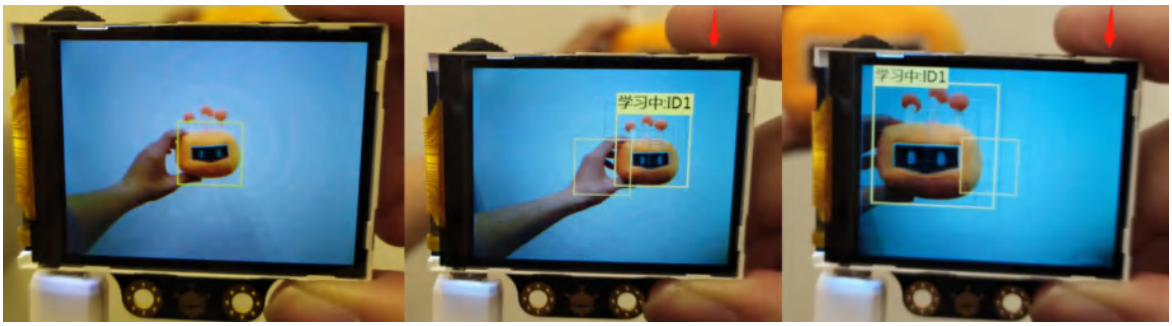
如果想要追光灯能够实时的跟随你的步伐移动，那就需要有一双“眼睛”紧盯你的步伐，并操作灯光实时追随着你，那我们如何实现这个功能呢？就要使用到我们使用的 HUSKYLENS 传感器中的物体追踪功能。

HUSKYLENS 传感器中的物体追踪功能是利用传感器内置算法，通过对物体的特征进行学习而实现对物体在屏幕中位置的追踪，并可以将位置的坐标值反馈给主控的功能。我们可以通过获取的物品位置值来驱动云台实现实时追踪的追光灯的功能。

学习物体：

与颜色识别或人脸识别不同，物体追踪是可以对一个物体（或人）进行完整的学习并进行识别的，颜色识别只针对颜色，而人脸则只是人体的一部分，物体追踪则是针对这个物体的整体特征进行学习从而进行追踪

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”。



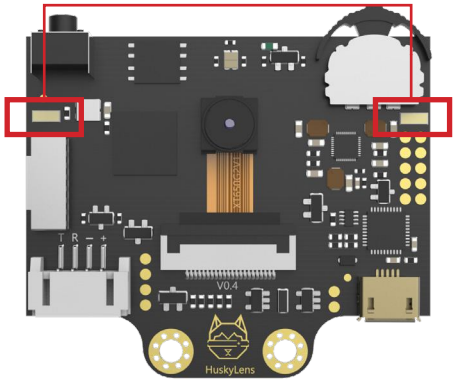
当 HuskyLens 在不同的角度和距离都能追踪到该物体时,就可以松开“学习按键”结束学习了。

> **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，请选择 “忘记学过的物品” 后重新学习。

打开 “追光灯”

HuskyLens 传感器上有两颗补光灯，可以让他在昏暗的环境下依然可以稳定的执行检测功能。

补光灯

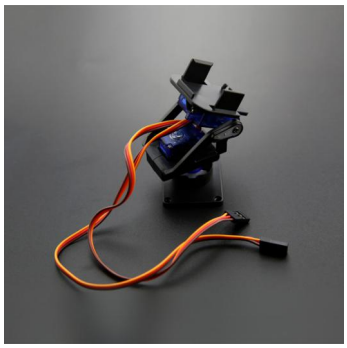


打开补光灯

切换到常规设置选项，进入设置菜单。在常规设置菜单下找到 LED 开关

摁下按钮就可以通过左右拨动设置 LED 灯的开关了。

三、2 自由度云台

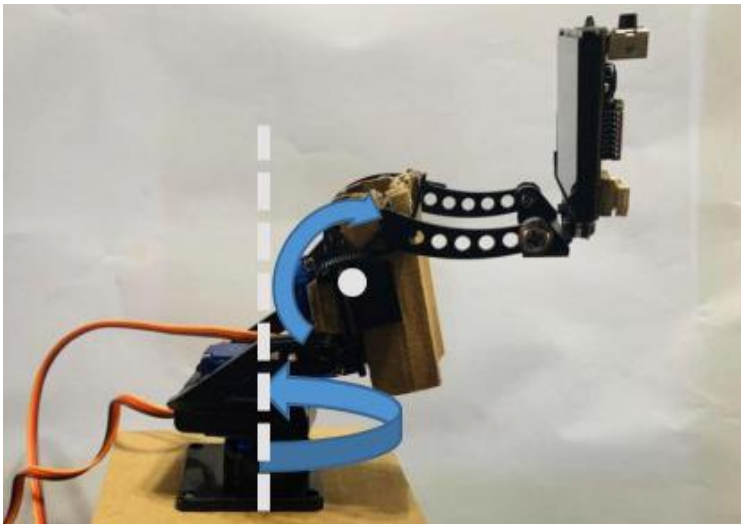


什么是云台?

云台是安装、固定摄像机的支撑设备，它分为固定和电动云台两种。固定云台适用于监视范围不大的情况，在固定云台上安装好摄像机后可调整摄像机的水平和俯仰的角度，达到最好的工作姿态后只要锁定调整机构就可以了。电动云台适用于对大范围进行扫描监视，它可以扩大摄像机的监视范围。电动云台高速姿态是由两台执行电动机来实现，电动机接受来自控制器的信号精确地运行定位。

云台在案例中的应用

二自由度云台的作用是移动摄像头使摄像头保持一直将识别的图框保持在识别区域中心的功能。这样我们的追光灯就可以保证一直照射在目标身上。二自由度云台的两个自由度分别是水平移动（X 轴）垂直移动（Y 轴）这两个方向的移动，分别对应我们云台上安装的两个舵机的移动方向。



什么是舵机?

舵机是一种可以指定控制位置（角度）的电机，可以通过 Mind+ 中的程序来指定控制舵机旋转的角度。我们最常用的舵机大多最大旋转角度是 0° ~180°，也有 90°或者其他角度的。也有比较特殊的 360°舵机，但是 360°舵机不能够控制其旋转到指定的角度。本节中我们使用的是 180°舵机。

项目实践:

我们将分为两步将任务完成，首先我们会学习使用 HuskyLens 的物体追踪功能，并读取物体的坐标数据。然后再次基础上添加云台实现最终的自动追光的功能。

任务一: 物体追踪与坐标值的作用

通过 HuskyLens 传感器获取到物体在屏幕中的位置坐标值，我们可以通过坐标值判断物体在传感器的相对位置。

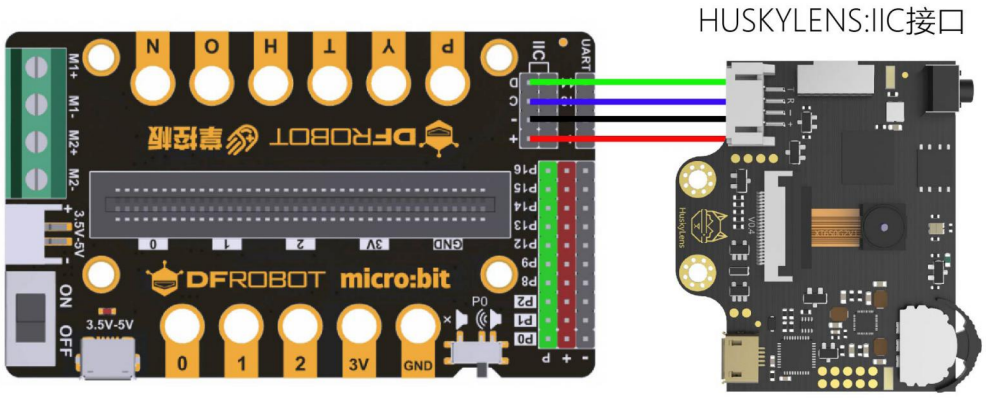
任务二: 云台驱动的追光灯

在上步获取了坐标值之后，驱动舵机移动 HuskyLens 传感器，让坐标值的位置移动到屏幕中心，这样就可以实现实时追光的效果。

任务一：物体追踪与坐标值的作用

1 硬件连接

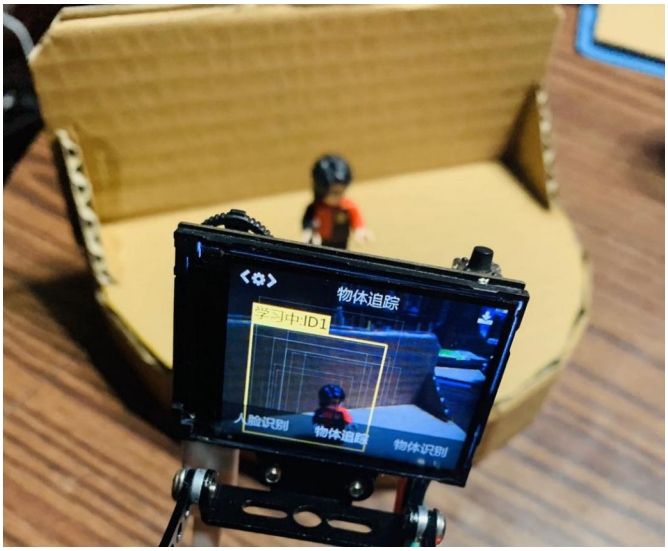
HuskyLens 传感器使用的是 IIC 接口，需要注意线序，不要接错或接反。



2 程序设计

STEP1 学习与识别

在设计程序之前我们需要让 HuskyLens 传感器学习我们需要追踪的“演员”。



STEP2 Mind+ 软件设置

打开 Mind+ 软件（1.62 或以上版本），切换到“上传模式”，点击“扩展”，在“主控板”下点击加载 “micro:bit”，在 “传感器” 下点击加载 “HUSKYLENS AI 摄像头”。

STEP3 指令学习

来认识一下主要用到的几条指令。

初始化，仅需执行一次，放在主程序开始和循环执行之间，可选择 I2C 或串口，I2C 地址不用变动。**注意 HUSKYLENS 端需要在设置中调整“输出协议”与程序中一致，否则读不出数据。**

切换算法，可以随时切换到其他算法，同时只能存在一个算法，注意切换算法需要一些时间。

主控板向 HUSKYLENS 请求一次数据存入“结果”（存在主控板的内存变量中，一次请求刷新一次存在内存中的数据），之后可以从“结果”中获取数据，此模块调用之后“结果”中才会获取到最新的数据。

从请求得到的“结果”中获取当前界面中是否有方框或箭头，包含已学习（id 大于 0）和未学习的，有一个及以上则返回 1。

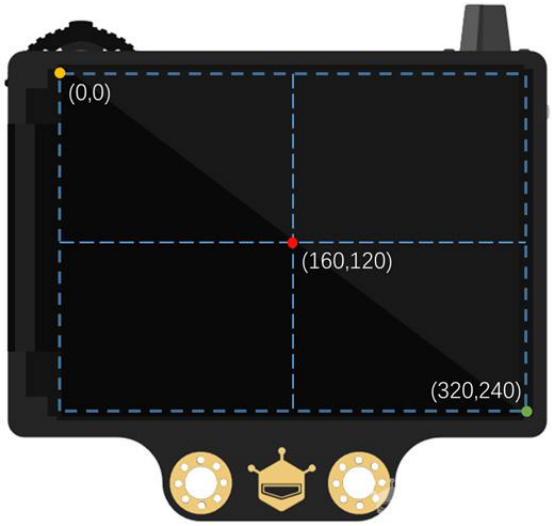
从请求得到的“结果”中获取是否 IDx 已经进行了学习

从请求得到的“结果”中获取是否 IDx 在画面中，方框指屏幕上目标为方框的的算法，箭头对应屏幕上目标为箭头的算法，当前仅为巡线算法时选择箭头，其他算法都选择方框。

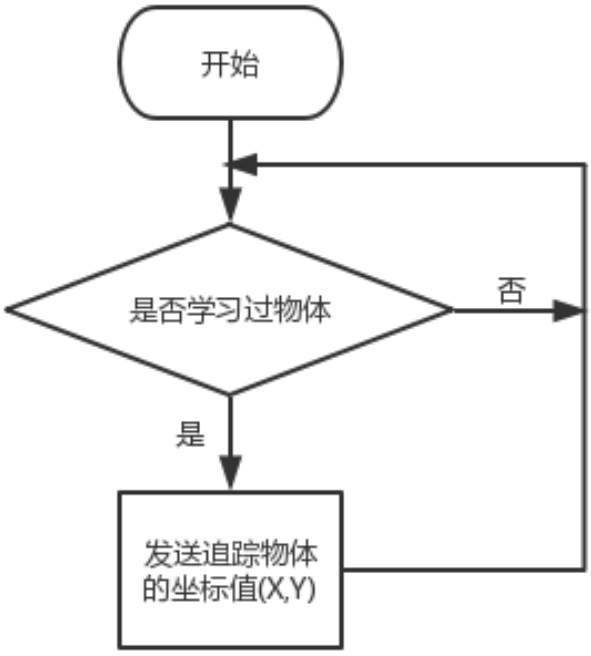
从请求得到的“结果”中获取是否 IDx 在画面中，方框指屏幕上目标为方框的的算法，箭头对应屏幕上目标为箭头的算法，当前仅为巡线算法时选择箭头，其他算法都选择方框。

STEP4 坐标分析

HuskyLens 传感器屏幕分辨率为 320*240，如图。那么我们通过程序获取的物体中心点坐标也就在这个范围之内。如，获取的坐标值为（160，120）那现在追踪的物体就在屏幕的中心。



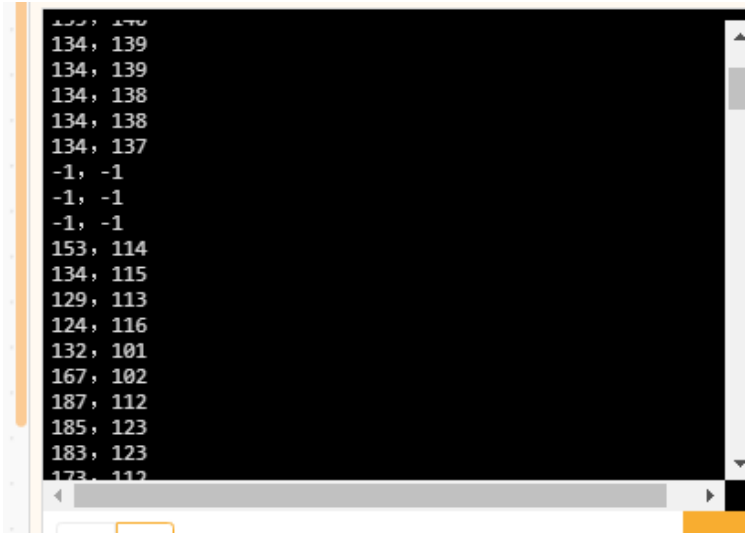
STEP5 流程图分析



3 程序示例

4 运行效果

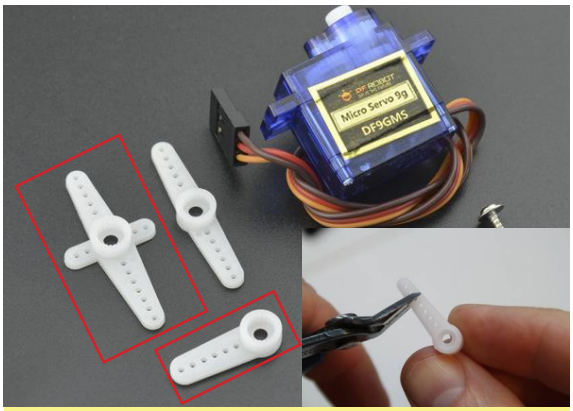
打开 Mind+ 中的串口，读取数据，出现 (-1, -1) 则表示追踪的物体丢失。



任务二：云台驱动的追光灯

1 结构搭建及硬件连接

在了解了物体追踪的功能和云台的功能后，我们就需要动手来组装了，首先我们组要组装二自由度云台。

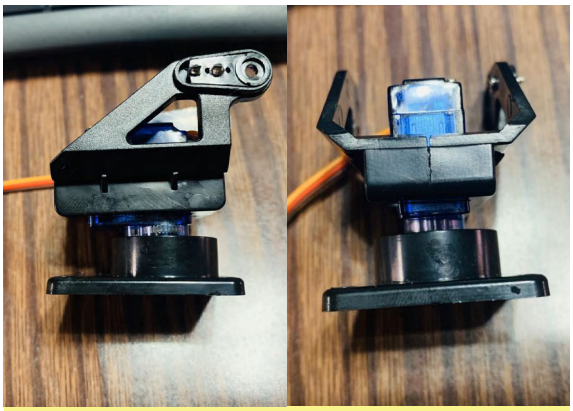


STEP1:
首先，找到舵机中的舵盘，进行修剪，需要修剪图中标注的两个。

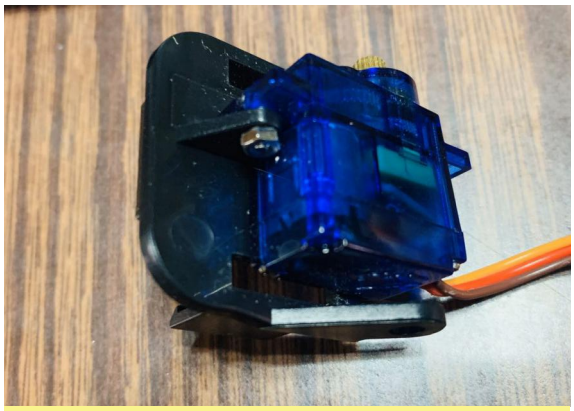
STEP2:
将修剪好的舵盘用螺丝固定在底座上

STEP3:
用固定座固定舵机，注意左右方向不要装反

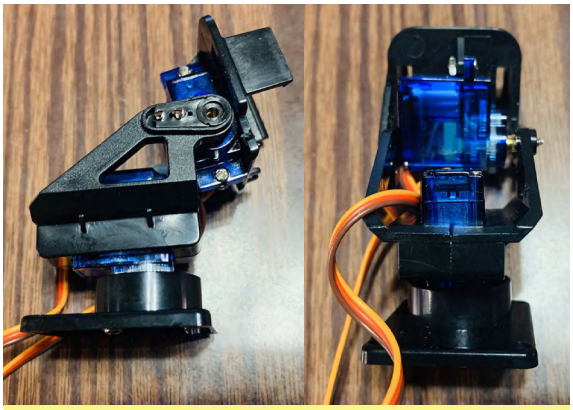
STEP4:
将修剪好的另一个舵盘用螺丝固定在图示位置



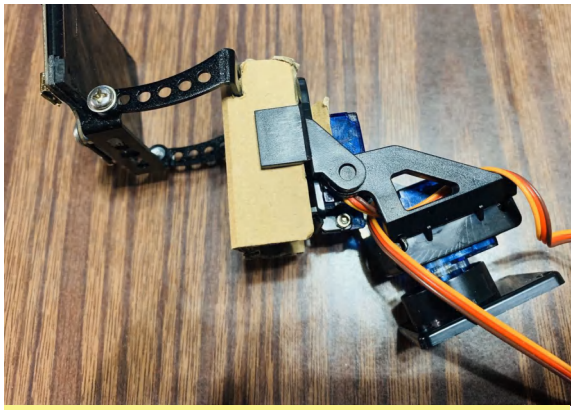
STEP5:
然后将底座与上步装好的组建组合



STEP6:
然后将最后一个结构件与舵机组装在一起

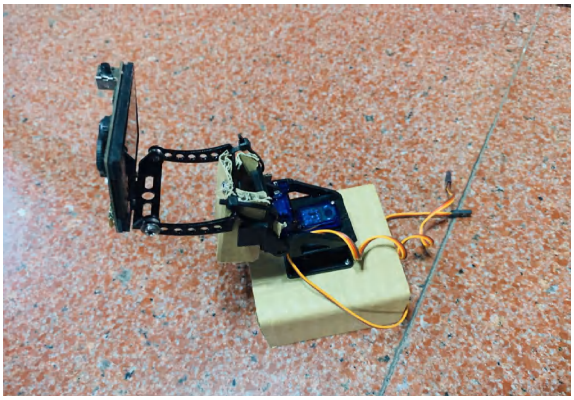


STEP7:
将装好的部件组装在一起就完成了，装入需要稍微掰开一点卡扣

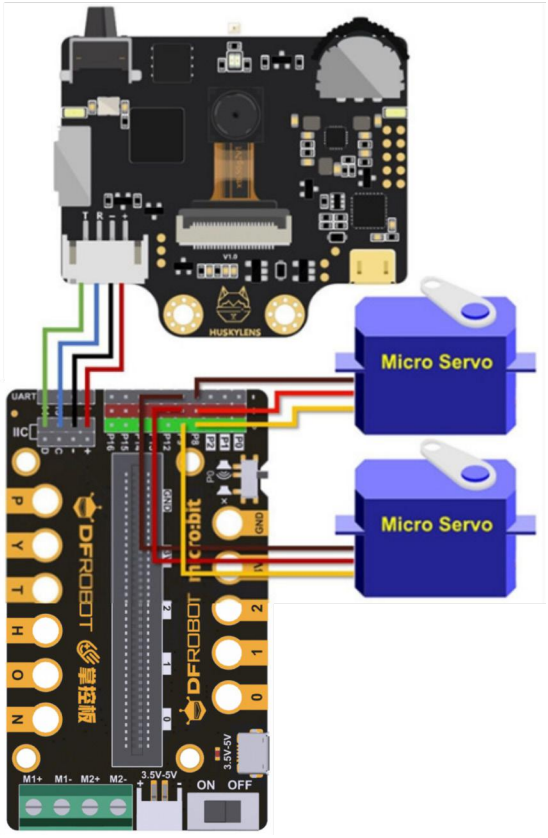


STEP8:
然后将 HuskyLens 传感器固定在上面，由于云台不是专为传感器设计所以我们可以简单添加一些结构来适配它。这里使用了一小片瓦楞纸来固定

STEP9:
装上 HuskyLens 传感器后，云台的重心就偏移了，导致云台无法立住，所以需要添加一个底座来固定它。先将底座固定在上面，然后再将其他部件组装在一起



完成后将他们的连接线接到扩展板上。



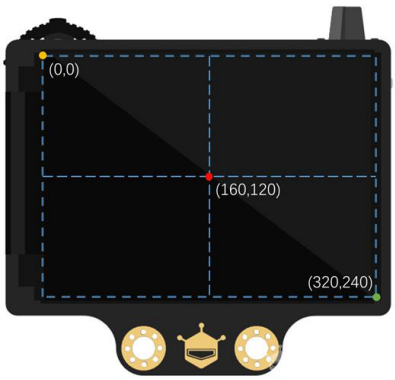
注: 接线时注意引脚一一对应

设备	扩展板引脚
HuskyLens	IIC
X 轴舵机	P8
Y 轴舵机	P9

2 程序设计

让追光灯保持追踪物体的话，主要需要的就是这个功能：

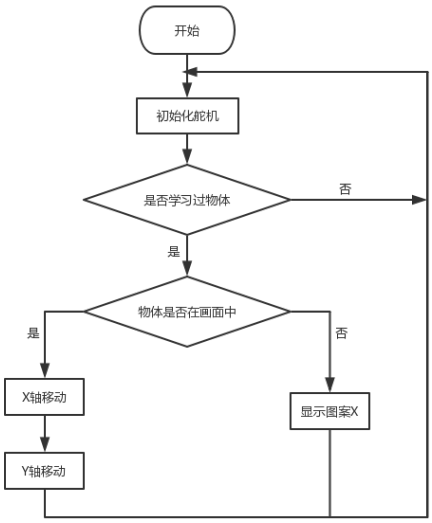
在上一步我们知道，HuskyLens 传感器的屏幕分辨率为 320*240，如图。那么屏幕的中心点坐标就是（160，120），那么我们可以根据对象方框的中心点坐标，让舵机作出以下动作，实现跟踪。



对象中心坐标	舵机动作
X<160	X 舵机向左
X>160	X 舵机向右
X=160	X 舵机停止
Y>120	Y 舵机向上
Y<120	Y 舵机向下
Y=120	Y 舵机停止

当对象方框中心点到达屏幕中心时，系统会判断为目标在追光灯的位置中心。追光灯就依靠哈士奇上自带的两颗 LED 来实现。

根据以上分析得出程序逻辑图如下：

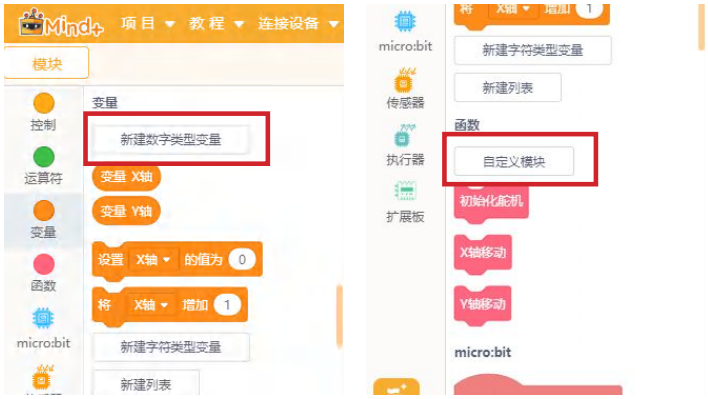


为了简化程序的复杂度，这里我们会用到变量和自定义函数的模块。变量的主要作用，就是用来存储信息，在之后需要使用的时候再去调用它。而函数则是存储一段指令，在之后需要的时候去调用它。

3 程序示例

为这段程序相交于上个案例多使用了两个舵机，所以需要在扩展模块的执行器中添加舵机模块。

1、首先我们设置变量和自定义函数，在新疆数字类型变量中新建 “X 轴”、“Y 轴” 这两个变量；在函数自定义模块中新建 “初始化舵机”、“X 轴移动”、“Y 轴移动” 这三个模块



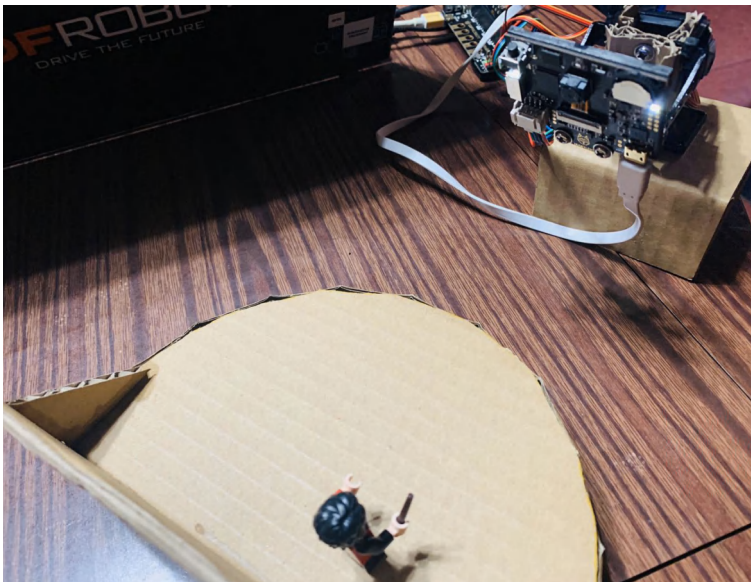
2、对三个自定义模块函数进行定义，首先是初始化舵机，目标是将舵机位置移动到初始值，并将舵机的旋转角度于变量 “X 轴” 和 “Y 轴进行了关联

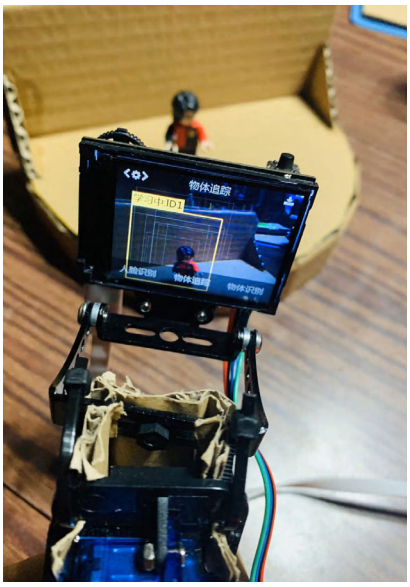
3、定义” X 轴移动 “，这里执行的功能是程序框图中对物体中心坐标 X 轴这部分判断执行

4. 同理，“Y 轴移动”的功能是依据程序框图里 Y 轴部分的判断

5. 最后我们来完成主程序，只需要调用之前编辑好的自定义函数就可以，程序内容简洁明了

4 运行效果





HuskyLens 传感器会根据舞台上人物的移动而跟随进行追光

项目小结:

项目回顾:

本节课学习使用了 HuskyLens 传感器的物体追踪功能，配合云台实现了自动追光灯的功能；在现实生活中，物体追踪功能还有这更多的应用。

知识点回顾:

1、学习了物体追踪的主要技术；

2、学习了云台和舵机的技术于使用；

3、学习了程序中变量和自定义函数的概念和使用。

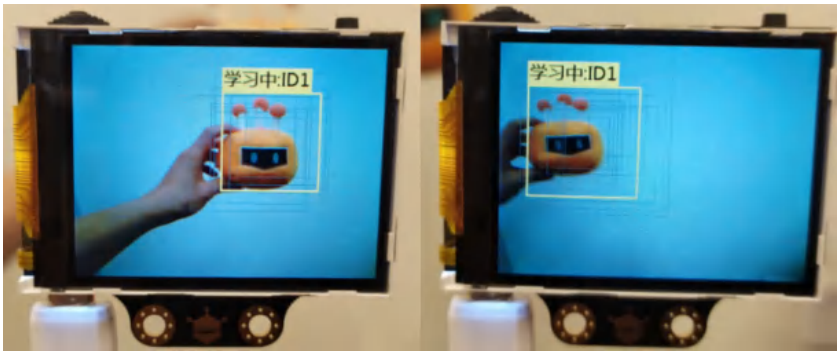
项目拓展:

在完成了智能追光灯项目之后，是否能够完成一个自动打靶装置？利用哈士奇的物体追踪功能来自动瞄准移动靶或固定靶。尝试使用激光灯来作为打靶的”子弹“，使用光线传感器来检测是否击中靶心的方式来实现功能。

拓展知识:

追踪物体深度学习:

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。



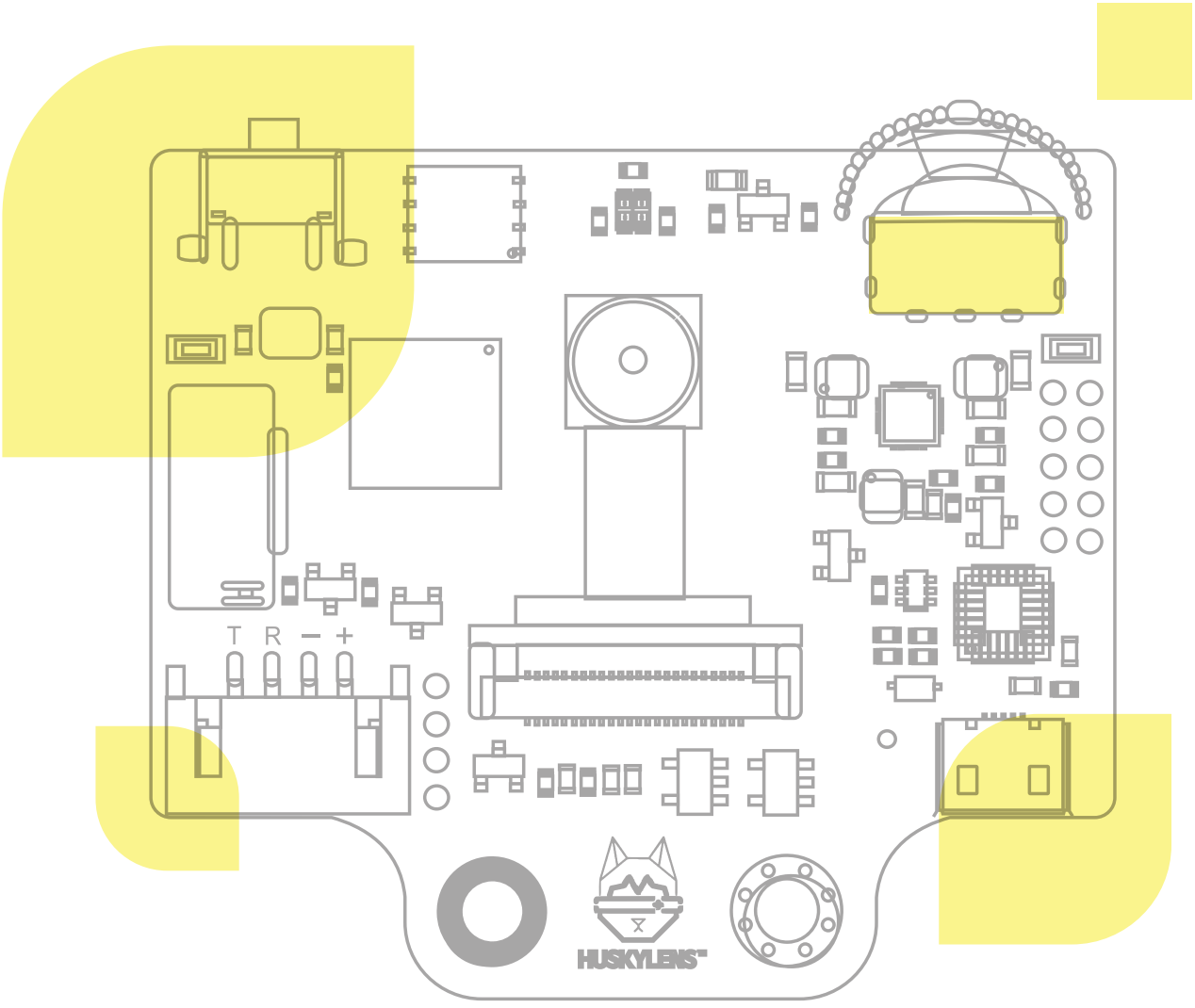
当识别结果满足要求,达到预期效果,就可关闭一边追踪一边学习的功能,方法是:长按 “功能按键”，进入物体追踪功能的二级菜单参数设置，选择 “学习开启”，关闭此参数即可（进度条颜色变白，进度条上的方块位于进度条的左边）。

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。

DFROBOT
DRIVE THE FUTURE

自助超市收银机

Project 4



如果你经常去超市购物，那么你可能在情不自禁地“剁手”后会遇到排队结算的窘境，当然，你买的东西越多，你就越囧。

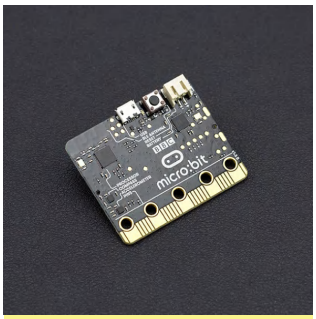
随着科技的发展，现在许多超市都有了自助收银机，它给我们的购物带来了方便。

现在我们就来做一个超市的自助收银机，等一等，我们需要怎样准确识别商品并结算呢？

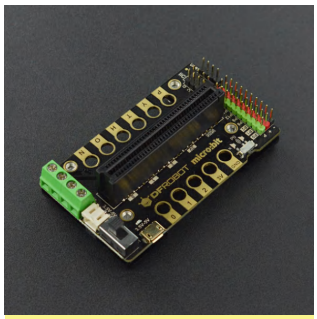
功能介绍:

本项目利用 HuskyLens 的标签识别功能，通过识别商品上特定的标签，实现计算总价的自助收银功能。

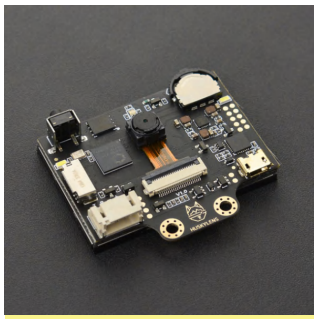
材料清单:



Micro:bit ×1



扩展板 ×1



HUSKYLENS ×1

知识园地:

我们仔细观察超市的收银过程，会发现无论是人工收银还是自助收银，都是通过扫码装置对商品的条形码进行扫码计费，每种商品的条形码都是不一样的，所以我们只需找到扫码装置和条形码的替代品就可以实现我们的项目了。

条形码 ------>AprilTag

扫码装置 ------>HUSKYLENS 的**标签识别功能**

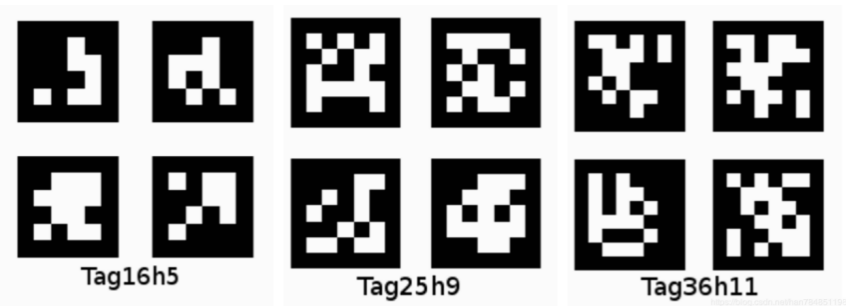
一、什么是标签识别?

标签识别技术（简称**标识技术**）是指对物品进行有效的、标准化的编码与标识的技术手段，它是信息化的基础工作。随着人们对于健康和安全的意识越来越强，食品行业对产品的质量和安全性（从原料，运输，到生产、贮藏以及涉及的追溯和管理）的要求越来越高、越来越多。标识在满足企业对产品追踪追溯需求等方面也起到了很重要的作用。

标识技术主要有条形码技术、IC 卡技术、射频识别技术、光符号识别技术、语音识别技术、生物计量识别技术、遥感遥测、机器人智能感知等技术。

二、什么是 AprilTag？

AprilTags 是一个出自密歇根大学项目团队的视觉基准系统，主要用于 AR，机器人和相机校准等领域。标签的作用类似于条形码，存储少量信息（标签 ID），同时还可以对标签进行简单而准确的 6D（x，y，z，滚动，俯仰，偏航）姿势估算。



三、HUSKYLENS 识别 AprilTag 原理

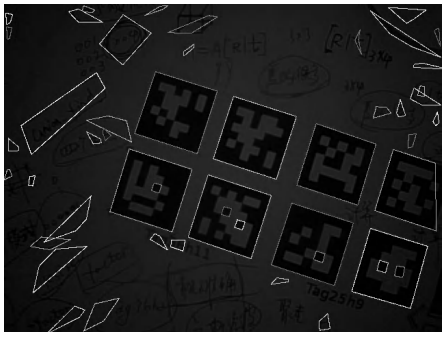
AprilTag 识别过程主要包含如下步骤：

1. 边缘检测

寻找图像中的边缘轮廓。

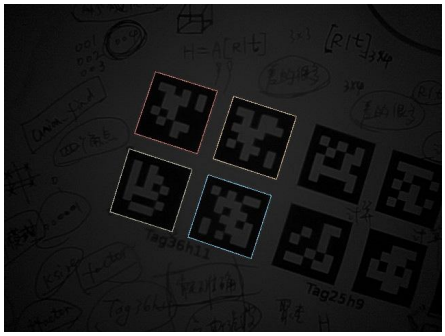
2、四边形检测

找出轮廓中的四边形。



3、解码

对找出的四边形进行匹配、检查。

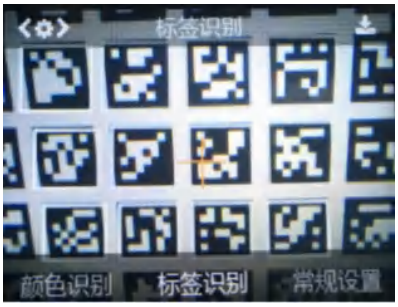


通过这些步骤，HUSKYLENS 的标签识别功能就能识别不同的 AprilTags，因此，我们只需要将不同的 AprilTags 贴在不同的商品上就可以实现识别对应商品了。

四、HUSKYLENS 传感器 - 标签识别功能演示

1、侦测标签

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。

本项目中，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。

项目实践：

我们将分三个递进任务来完成项目，首先我们使用 HuskyLens 的标签识别功能学习并识别贴有 AprilTags 的三件商品；然后在前一步基础上添加开始扫码和结束扫码的事件，方便不同顾客的商品统计；最后在第二步基础上添加商品总价结算功能，实现现实生活中的超市自助收银功能。

任务一：识别商品

HuskyLens 学习并识别贴在三个不同商品上的标签，编写程序让 Micro:bit 的点阵滚动显示对应的商品名。

任务二：开始与结束扫码

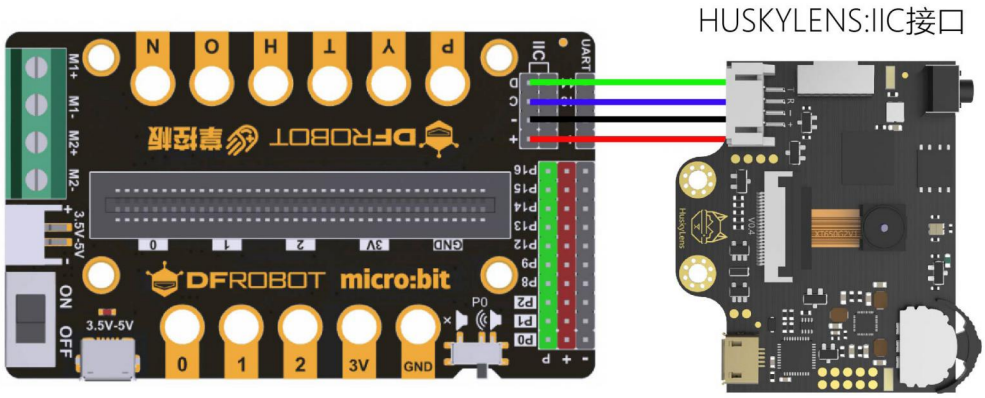
顾客按下 Micro:bit 的 A 键，点阵依次滚动显示 HuskyLens 识别到的商品名，按下 B 键，点阵不再显示该顾客扫描到的任何商品，下一个顾客按下 A 键重新开始扫描。

任务三：商品结算

在任务二基础上，当识别到任意学习过的商品时，都会在总价中将其售价加入，按下 B 键结束扫码时，点阵会显示顾客从开始扫描后到结束扫描前的所有商品的总价。

任务一：识别商品

1 硬件连接



2 程序设计

STEP1 学习与识别

假设超市有且仅有水杯、酥饼、美工刀三种商品，它们分别对应三个不同的标签。

HuskyLens 能识别这 3 个不同标签，在“多次学习”模式下按水杯、酥饼、美工刀的顺序完成学习，得到 ID1、ID2、ID3，如果下次识别到相同的标签，HuskyLens 就会返回对应的 ID，这样我们就可以使用选择结构让点阵屏显示 ID 对应的商品名。

STEP2 Mind+ 软件设置

打开 Mind+ 软件（1.62 或以上版本），切换到“上传模式”，点击“扩展”，在“主控板”下点击加载 “micro:bit”，在 “传感器” 下点击加载 “HUSKYLENS AI 摄像头”。

STEP3 指令学习

来认识一下主要用到的几条指令。

初始化，仅需执行一次，放在主程序开始和循环执行之间，可选择 I2C 或串口,I2C 地址不用变动。**注意 HUSKYLENS 端需要在设置中调整“输出协议”与程序中一致，否则读不出数据。**

切换算法，可以随时切换到其他算法，同时只能存在一个算法，注意切换算法需要一些时间。

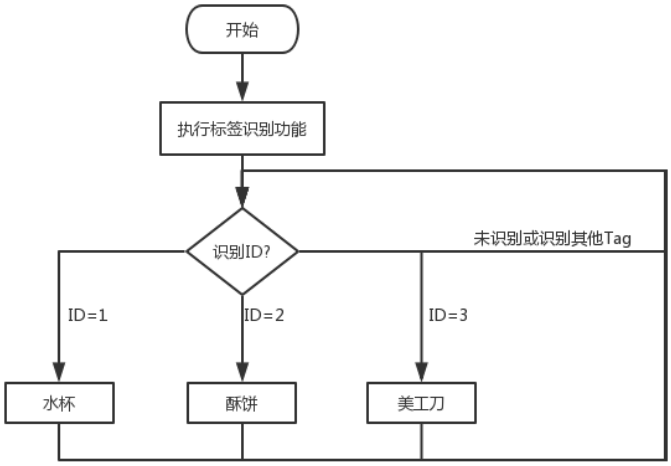
主控板向 HUSKYLENS 请求一次数据存入“结果”（存在主控板的内存变量中，一次请求刷新一次存在内存中的数据），之后可以从“结果”中获取数据，此模块调用之后“结果”中才会获取到最新的数据。

从请求得到的“结果”中获取当前界面中是否有方框或箭头，包含已学习(id 大于 0)和未学习的，有一个及以上则返回 1。

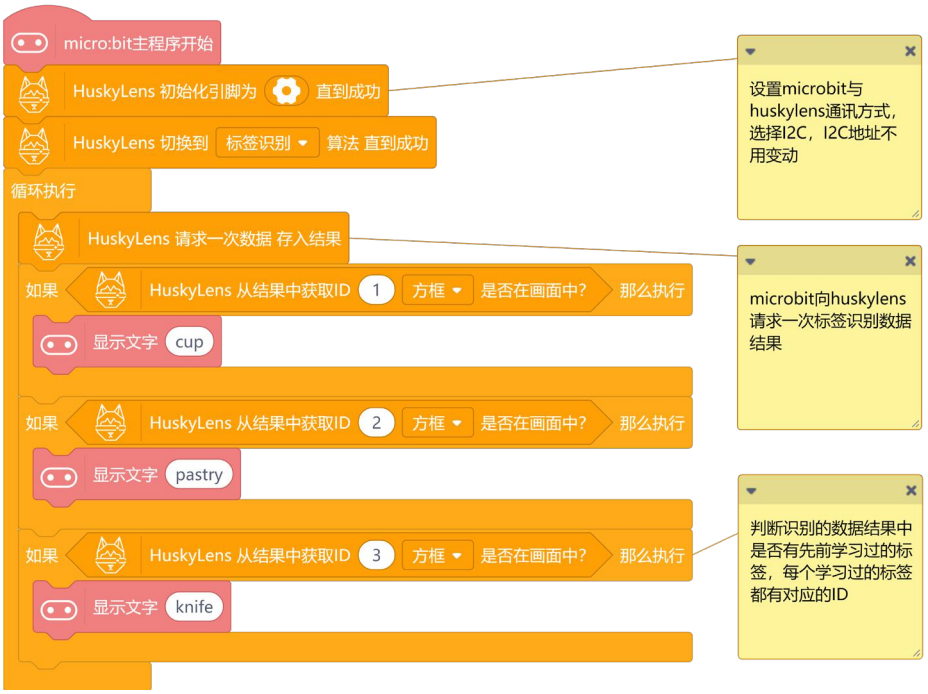
从请求得到的“结果”中获取是否 IDx 已经进行了学习

从请求得到的“结果”中获取是否 IDx 在画面中，方框指屏幕上目标为方框的的算法，箭头对应屏幕上目标为箭头的算法，当前仅为巡线算法时选择箭头，其他算法都选择方框。

STEP4 流程图分析



3 程序示例



4 运行效果

当识别到水杯、酥饼、美工刀时，点阵会滚动显示“cup”“pastry”“knife”，当没有标签或识别到其他标签，点阵屏会一直处于熄灭状态。

任务二：开始与结束扫码

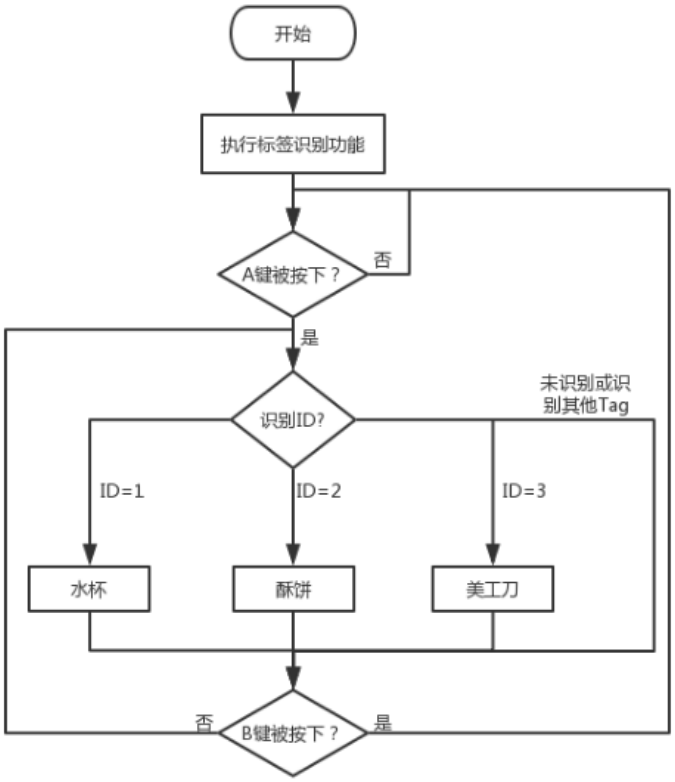
1 硬件连接

同任务一。

2 程序设计

我们需要添加事件让顾客知道何时开始扫描商品，何时结束。假设按下 Micro:bit 上的 A 键开始扫描商品，每一个商品信息滚动显示完就可以进行下一个商品的扫描，那么扫描商品的过程就是一个循环，而跳出这个循环的条件就是按下 B 键，下一个顾客需要扫描商品时，只需再按一下 A 键即可。

根据以上分析得出程序逻辑图如下：



3 程序示例

4 运行效果

按下 A 键前，点阵不会显示扫码识别到的商品名；

按下 A 键后、按下 B 键前，点阵依次显示扫码识别到的商品名；

按下 B 键后，点阵不会显示扫码识别到的商品名。

任务三：商品结算

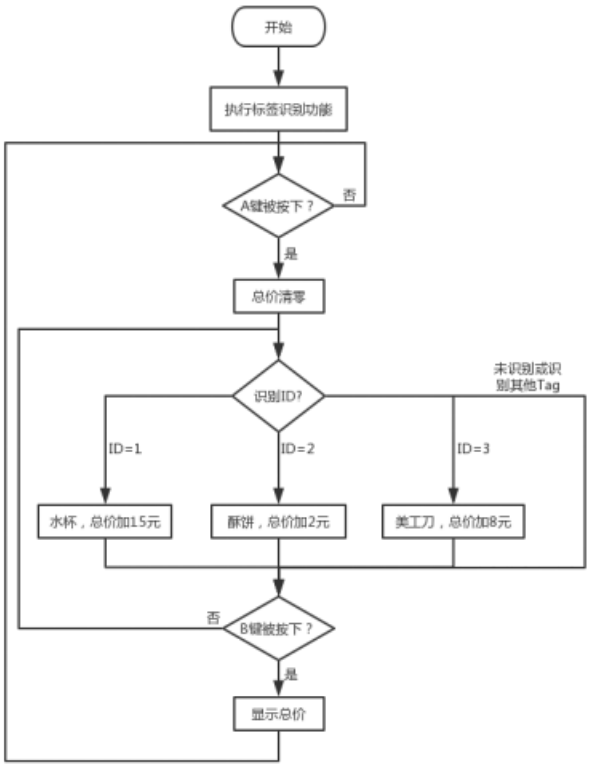
1 硬件连接

同任务一。

2 程序设计

商品的总价会随着扫描的商品的数量增加而变大，所以只需在任务二的基础上添加一个变量即可,每次按下A键后都需要将前一个顾客的总价清零,识别到一个商品，便将该商品对应的价格加入总价中，按下 B 键后显示所有商品的总价。

根据以上分析得出程序逻辑图如下:



3 程序示例

4 运行效果

在任务二的运行效果基础上，按下 B 键后，点阵会显示总价。

项目小结：

项目回顾：

本项目主要使用二维码标签表示商品的信息，通过标签识别输出特定的 ID，从而实现让点阵显示对应商品信息的功能，并添加了自动结算总账功能，基本上完成了超市的自助收银系统。

知识点回顾：

1、认识标签识别在生活中的重要性和使用方法

2、学习识别标签并做出相应的判断

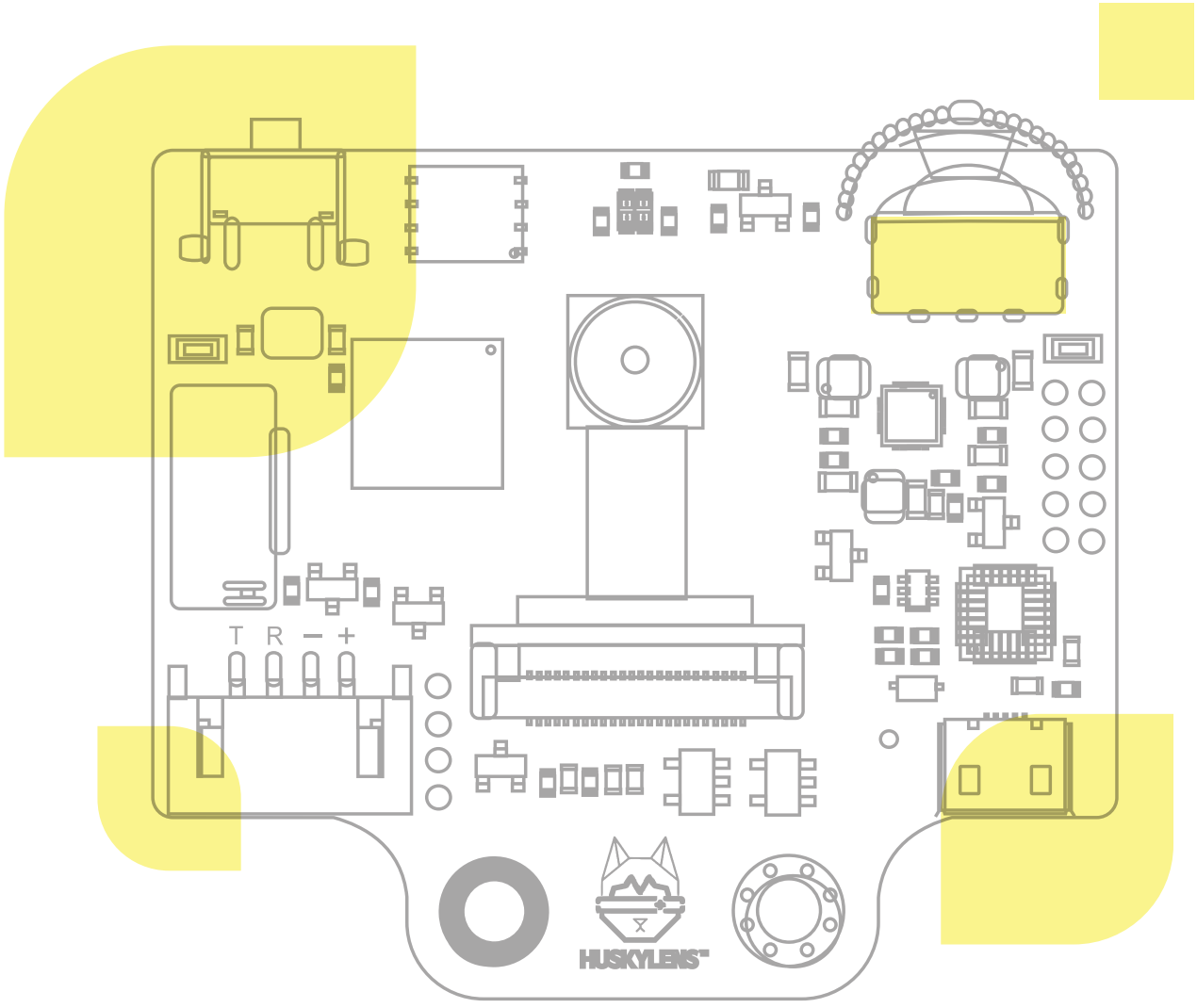
项目拓展：

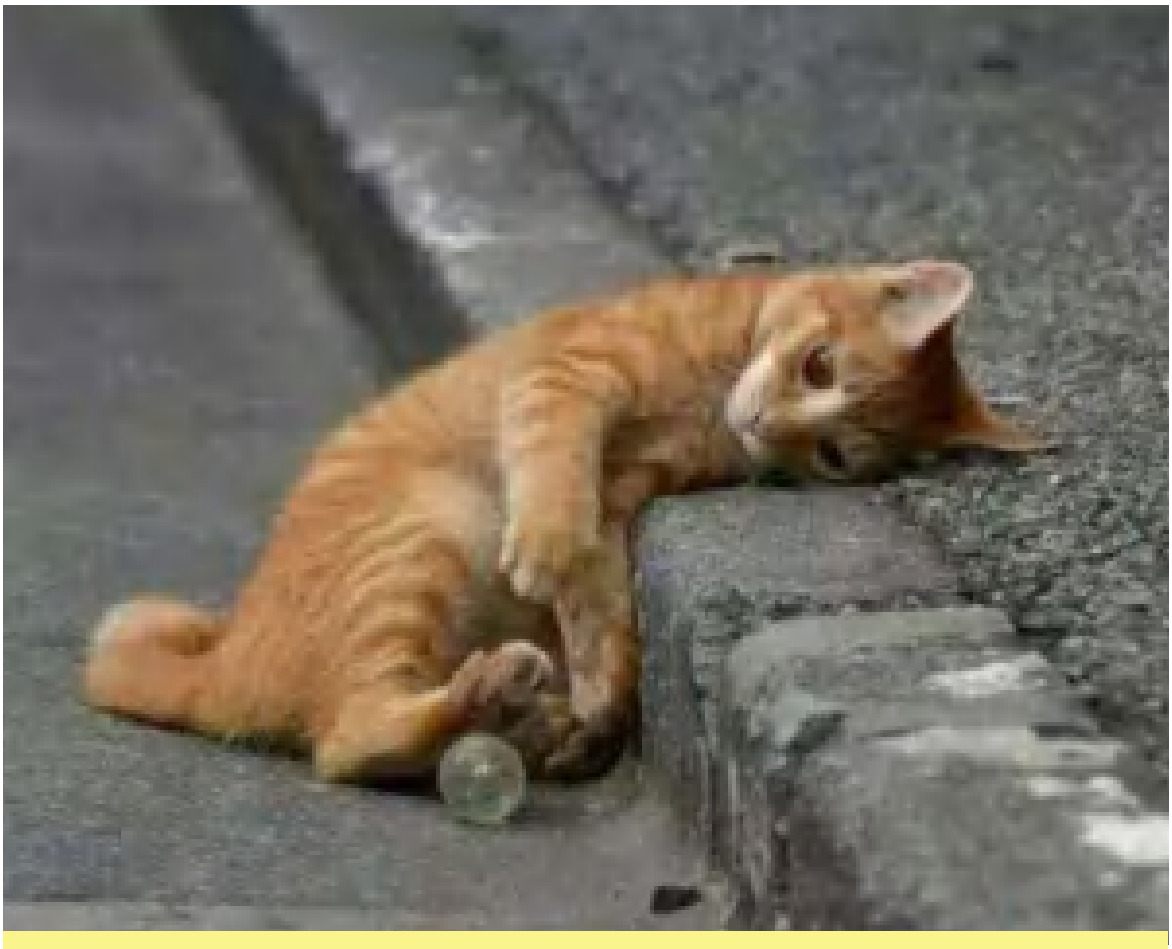
在完成了自助收银之后，我们可否用标签表示一个房屋位置信息，从而让扫地机器人在行进过程中及时判断自己的方位并做出相应调整？

DFROBOT
DRIVE THE FUTURE

流浪猫狗喂食机

Project 5





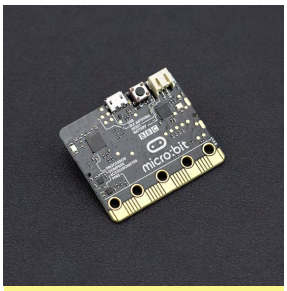
现在人们的生活条件好了，看到好看的小猫小狗就头脑一热买来饲养，却又没做好当一个毛孩子监护人的各种准备，结局便是残忍地遗弃它们。正因为有这些不负责任的猫狗饲主，垃圾站边上、路边的草丛甚至汽车的车底我们经常能看见这些可怜的小家伙，我们是否可以为它们做些什么？

如果有一台喂食机，用 HuskyLens 摄像头判断前方的是小猫还是小狗，为它们提供对应的猫粮或狗粮，是不是像一台流浪界的扫脸自动售货机，能够解决它们的温饱。之前被半吊子的主人伤害，现在至少能被人工智能温柔以待。

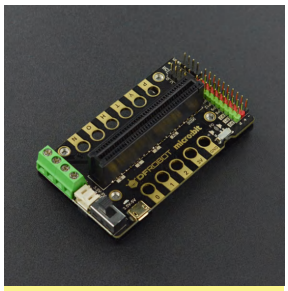
功能介绍:

本项目结构部分采用快递纸盒为框架搭建，两个塑料瓶保留了螺旋紧固结构便于补充猫狗粮，电子部分利用 HuskyLens 的物体识别功能，通过机器学习来区分猫和狗，由 micro:bit 板处理结果并控制舵机，打开的阀门，为面前的流浪猫狗出粮。

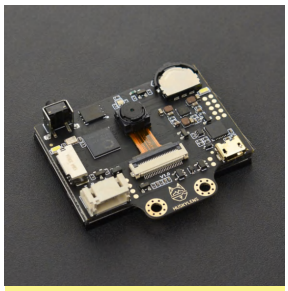
材料清单:



Micro:bit ×1



扩展板 ×1



HUSKYLENS ×1

DF9GMS 180° 微型舵机 x1

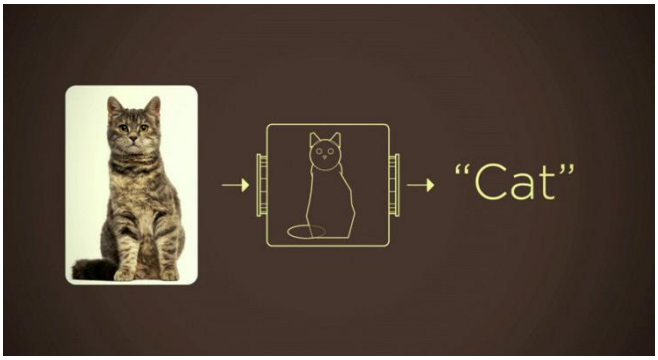
知识园地:

图像识别技术是人工智能的一个重要领域。它是指对图像进行对象识别，以识别各种不同模式的目标和对像的技术。而我们这个项目就是借助 HUSKYLENS 传感器的**图像识别**功能来对猫狗进行区分和识别。

一、什么是图像识别?

图像识别，是指利用计算机对图像进行处理、分析和理解，以识别各种不同模式的目标和对象的技术，是应用深度学习算法的一种实践应用。现阶段图像识别技术一

般分为人脸识别与商品识别，人脸识别主要运用在安全检查、身份核验与移动支付中；商品识别主要运用在商品流通过程中，特别是无人货架、智能零售柜等无人零售领域。



图像识别工作原理：

图像的传统识别流程分为四个步骤：

图像采集：顾名思义，通过摄像头采集图像，为了后面的识别工作做准备。

图像预处理：经过一系列算法，对图像中的一些信息进行分析和处理。

特征提取：根据上步处理过的信息，在其中提取关键信息，如：颜色、外轮廓等等。

图像识别：将提取的信息与样本库中的内容进行比对，在 HUSKYLENS 传感器的图像识别中既包含内置的样本库还可以通过学习来丰富样本库。

图像识别技术与其他识别技术的异同：

到现在我们已经学了许多摄像头识别的功能，如人脸识别、颜色识别等等，那他们之间有什么区别呢？

首先是人脸识别，人脸识别也是图像识别的一种，我们可以这么理解，人脸识别是专门用于区分人脸的图像识别。在场景中就就是：一群人经过人脸识别摄像头，如果提前录入过数据，它就能精确的“叫出”每一个人的名字，而图像识别功能得出的结

果就是：人、人、人。因为它只能识别物体的类别而不能对个体进行区别。

那我们就想到，那图像识别和物体追踪是不是有点像呢？都是识别物体的功能，但是细心观察就能发现，物体追踪功能只能学习并追踪单一一个物体，而图像识别可以识别多种物体。这是因为，物体追踪可以对一个物体的多个角度进行学习，当你让 HUSKYLENS 传感器学习物体的时候缓慢旋转角度就能够学习这个物体的各个角度的样子，这样就可以非常精准的追踪，而物体识别是学习物体的一个面，当你旋转物体的时候就无法识别了。

颜色识别和二维码识别则是对某些功能定向的识别功能，相信大家不会搞混。

图像识别技术主要应用在以下领域：

1. 生物医学：

图像识别在现代医学中的应用非常广泛，它具有直观、无创伤、安全方便等特点。在临床诊断和病理研究中广泛借助图像识别技术，例如在新冠肺炎期间，就出动人工智能来进行快速审阅病人 CT 的应用。

2. 遥感图像识别：

航空遥感和卫星遥感图像通常用图像识别技术进行加工以便提取有用的信息。该技术主要用于地形地质探查，森林、水利、海洋、农业等资源调查，灾害预测，环境污染监测，气象卫星云图处理以及地面军事目标识别等。

二、HUSKYLENS 中的物体识别

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。

1. 操作设置

向左或向右拨动“功能按键”，直至屏幕顶部显示“物体识别”。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。

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。

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。

2. 侦测物体

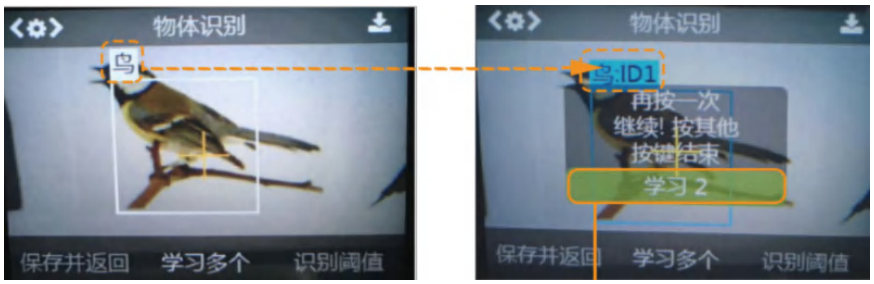
把HuskyLens对准目标物体在屏幕上会有白色框自动框选出识别到的所有物体，并显示对应的物体名称。目前只能识别并框选 20 种物体，其余物体无法识别和框选。

3. 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。



4. 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,并且做追踪。

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。

项目实践:

我们将分为两步将任务完成，首先我们会学习使用 HuskyLens 的图像识别功能，并将识别到的结果输出。我们可以根据靠近的是猫或者狗来给它们分发对应的粮食。

任务一: 区分猫狗

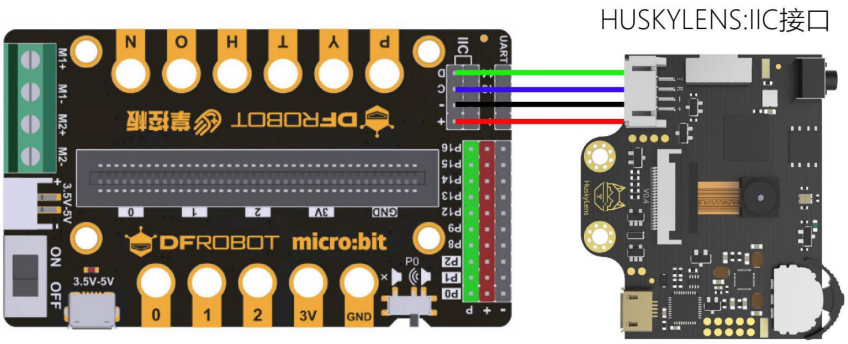
在这步我们需要让 HuskyLens 摄像头能够识别并区分猫和狗并能给出反馈，以便我们下一步能够实现发出对应的食粮。

任务二: 添加发粮功能

这步就需要在上步的基础上添加发粮功能，并制作对应的结构。

任务一: 区分猫狗

1 硬件连接



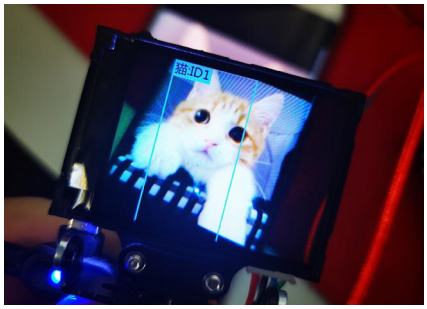
HuskyLens传感器使用的是IIC接口需要注意线序,不要接错或接反。

2 程序设计

这里我们需要让 HuskyLens 传感器学习猫和狗的图像，并能够在识别后输出猫和狗的指示。

STEP1 学习与识别

在设计程序之前我们需要让 HuskyLens 传感器学习猫和狗的图像。(注意需要先开启学习多个的功能)



STEP2 Mind+ 软件设置

打开 Mind+ 软件(1.62 或以上版本),切换到“上传模式”,点击“扩展”,在“主控板”下点击加载 “micro:bit”,在 “传感器” 下点击加载 “HUSKYLENS AI 摄像头”。

STEP3 指令学习

来认识一下主要用到的几条指令。

初始化，仅需执行一次，放在主程序开始和循环执行之间，可选择 I2C 或串口,I2C 地址不用变动。**注意 HUSKYLENS 端需要在设置中调整“输出协议”与程序中一致，否则读不出数据。**

切换算法，可以随时切换到其他算法，同时只能存在一个算法，注意切换算法需要一些时间。

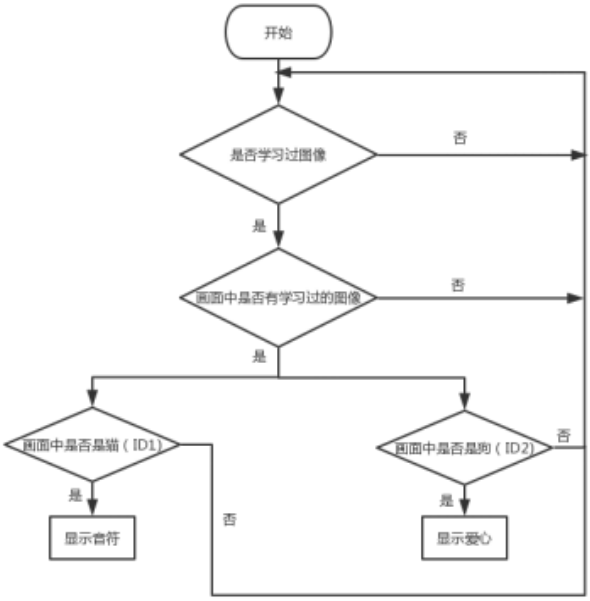
主控板向 HUSKYLENS 请求一次数据存入“结果”（存在主控板的内存变量中，一次请求刷新一次存在内存中的数据），之后可以从“结果”中获取数据，此模块调用之后“结果”中才会获取到最新的数据。

从请求得到的“结果”中获取当前界面中是否有方框或箭头，包含已学习(id 大于 0)和未学习的，有一个及以上则返回 1。

从请求得到的“结果”中获取是否 IDx 已经进行了学习

从请求得到的“结果”中获取是否 IDx 在画面中，方框指屏幕上目标为方框的的算法，箭头对应屏幕上目标为箭头的算法，当前仅为巡线算法时选择箭头，其他算法都选择方框。

STEP4 流程图分析



3 程序示例

4 运行效果

当在HUSKYLENS传感器中识别到猫会显示音符图形识别到狗会显示爱心图形。

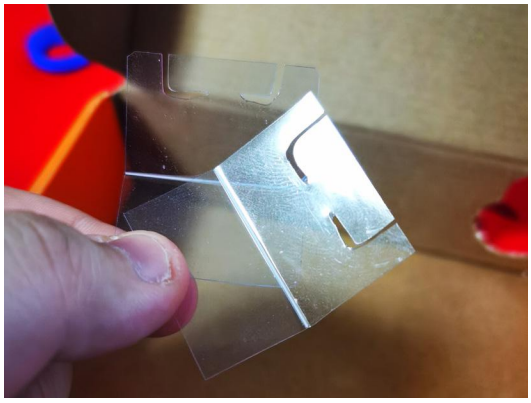
任务二：添加发粮功能

1 结构搭建

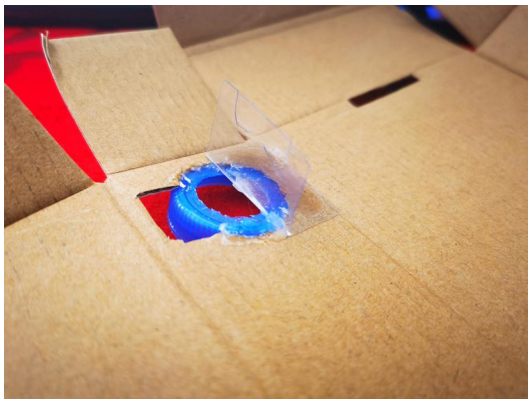
首先我选择了一个方盒子作为喂食机的主体，结构简单且牢固，还有盖子便于开合内部检修，相当适合这个项目。

取用两个塑料瓶的瓶盖开孔，如图用胶固定在快递盒顶部用剪刀剪出来的洞上，左右对称安装。

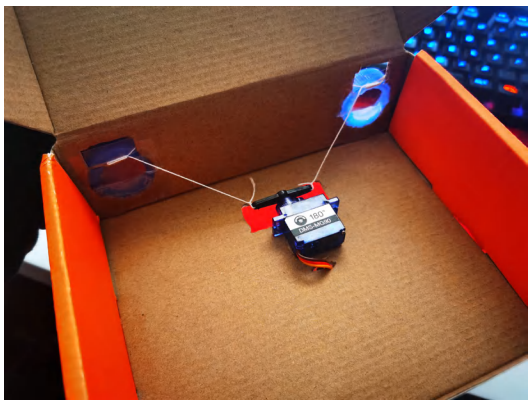
找两个塑料片，对折做出合页的效果，在一端剪出槽是为了之后更方便也更稳定的绑线。



将做好的阀门从盒子内部用胶固定，左右对称，确保合页打开时有够大的孔出猫狗粮。



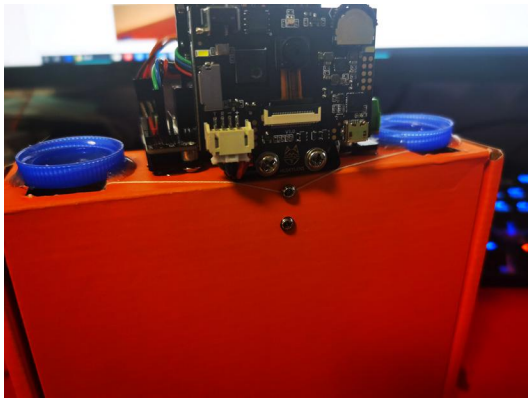
这里是很关键的一步，先用胶固定住舵机，在快递盒背面给摇臂开两个孔，一个给摇臂旋转空间还一个走舵机线，这里记得校准舵机的 90 度位置。再用一根线绑住摇臂左端，缠绕在左侧的塑料片上，然后绳子穿越飞机盒外侧从右侧的孔绕回来缠绕在右侧的塑料片上，最后绑到舵机摇臂的右端。



快递盒内部用硬纸片搭建出两侧猫粮和狗粮滚落的轨道。

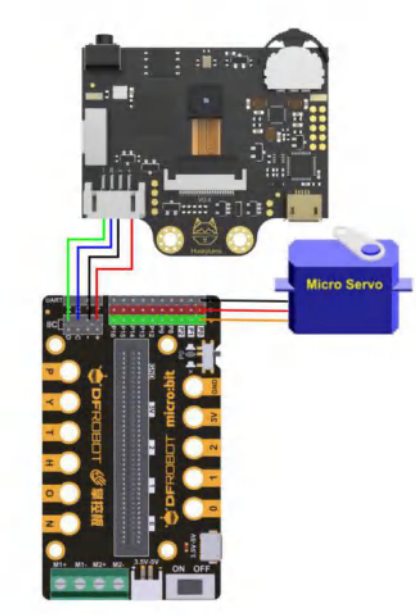
在快递盒盖的对应位置开孔让猫狗粮能够顺利滚落。

在顶部顺序固定好拓展板、micro:bit 板和哈士奇之后，在盒子正面拧上螺丝，是为了调节盒子外的绳子的松紧度，也间接地调整了两个阀门的松紧，我们可以根据猫狗粮不同的颗粒大小，来适当调节这里绳子的松紧。



到这里结构就先告一段落，我们等电子部分调试好之后再加上储存猫狗粮的塑料瓶。

2 硬件连接

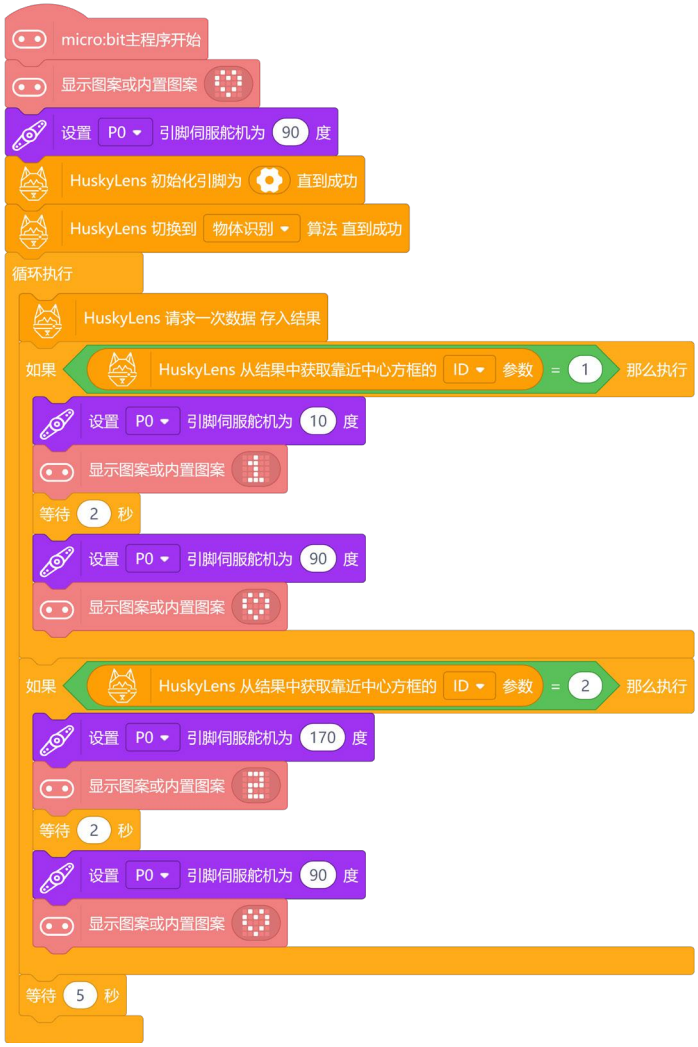


这里我们将 HuskyLens 摄像头通过 4pin 连接线连接到拓展板的 I2C 接口，作为整个系统的输入设备，接着将舵机按照正确的线序连接到拓展板的 P0 口。

3 程序实现

然后我们将 micro:bit 板接上电脑打开 Mind+ 开始程序的编写。首先完成 micro:bit 板驱动舵机的测试，确认阀门的开启与关闭，适当调整绳子的松紧度。

最后的主程序就简单许多，开机初始化一下 LED 点阵、舵机中位和 HuskyLens 摄像头，每隔一定的时间周期识别一次，识别到猫和狗舵机打开不同的阀门，掉落对应的猫粮或狗粮。



4 运行效果

当识别到猫靠近时，点阵屏显示 1，并投放猫粮，投放成功后显示爱心。当识别到狗靠近时，点阵屏显示 2，并投放狗粮，投放成功后显示爱心。

项目小结：

项目回顾：

我们用生活中随处可见的材料结合 HuskyLens 摄像头一步一步完成了充满温馨

的流浪猫狗喂食机，不仅仅是对流浪猫狗的生存福音，更是方便了有爱心的人自主添加猫狗粮，献出自己的爱心来帮助它们。

知识点回顾：

1、了解物体识别的工作原理

2、学习了 HuskyLens 物体识别的学习过程

3、能够利用 HuskyLens 作为输入设备结合 micro:bit 板和其他硬件完成项目

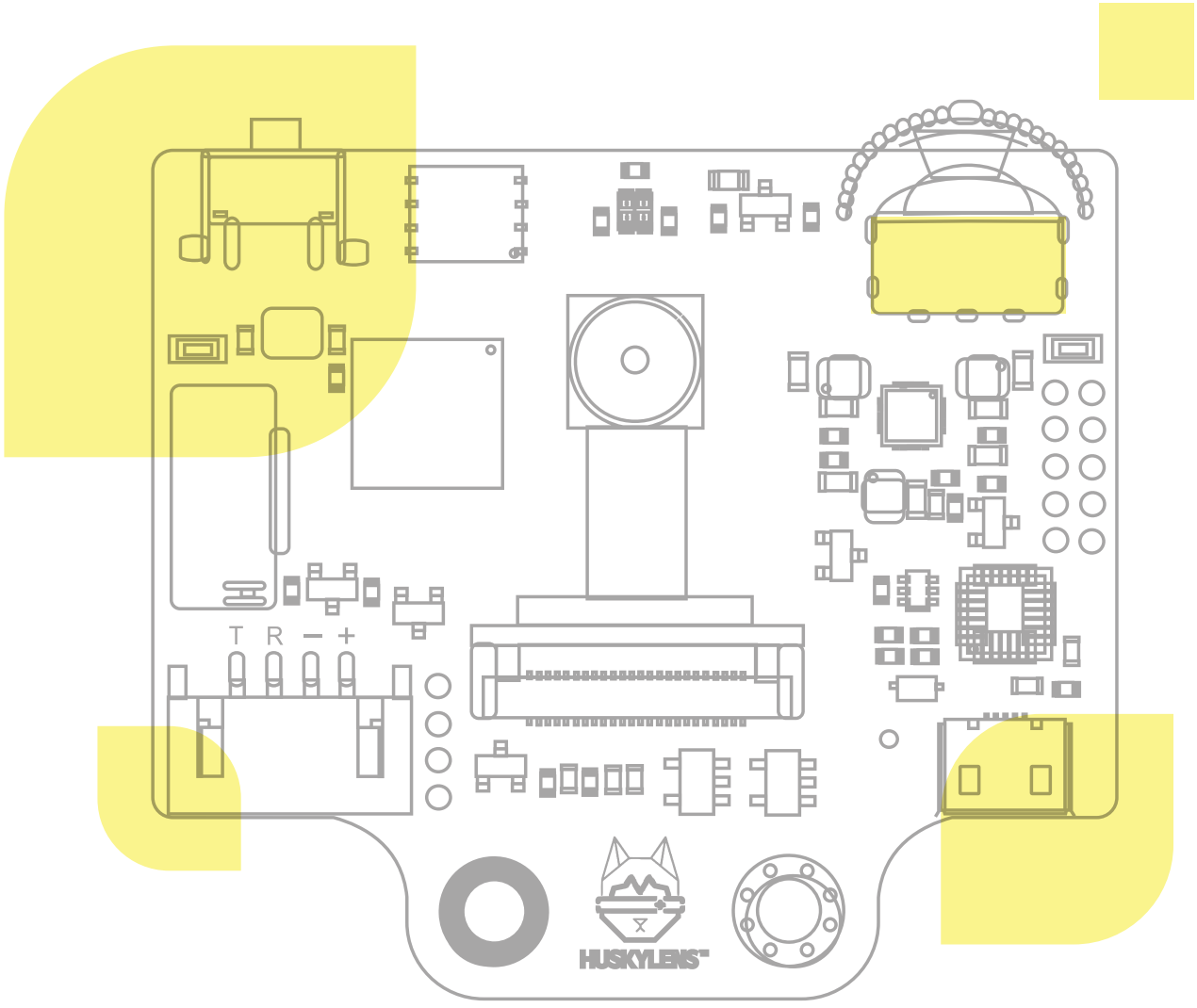
项目拓展：

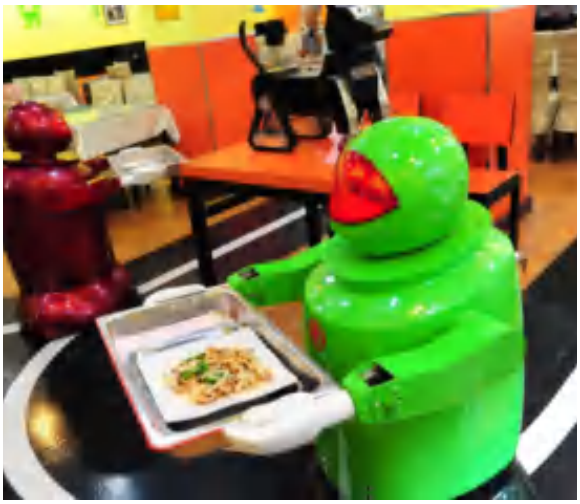
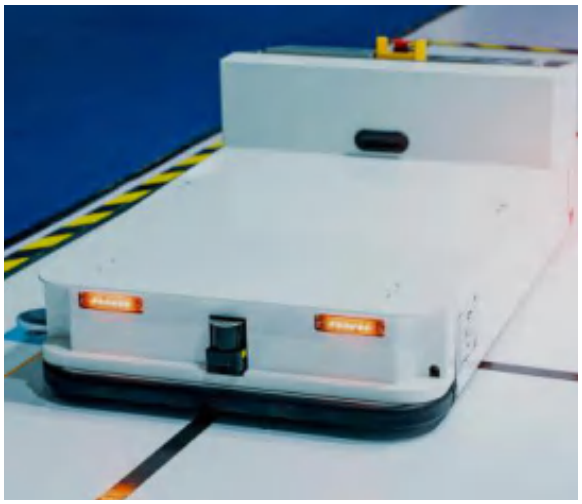
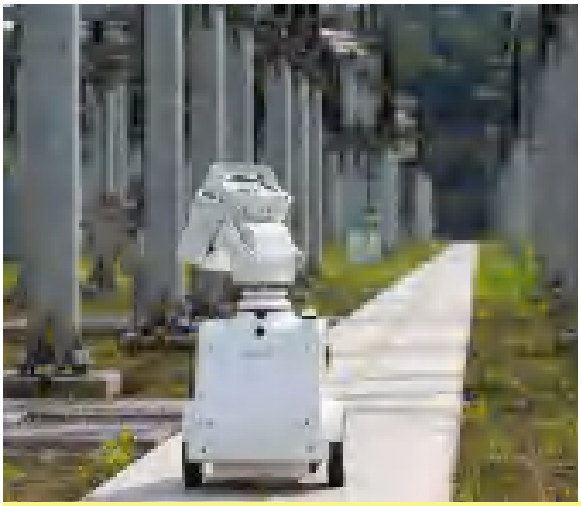
完成了流浪猫狗喂食机后，如果想要更深入的帮助流浪猫狗，是否可以添加一个统计的功能，记录每天来喂食机吃东西的猫和狗的数量，这样就能够更合理的配给猫粮和狗粮的量。

DFROBOT
DRIVE THE FUTURE

循“轨”蹈矩的麦昆

Project 6




如今的机器人真是越来越厉害了，它们上天入地，几乎无所不能。你看它们有的在餐厅当送餐员，服务周到；有的在工厂车间当“快递小哥”，任劳任怨；有的在电网枢纽当安全检查员，尽职尽责 ...

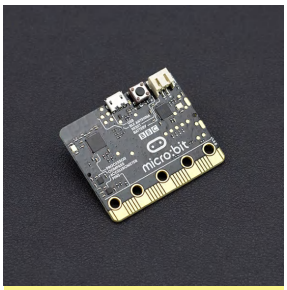
如果大家仔细观察，会发现这些机器人工作的地面上有线条，它们就是循着线条行进的。

巡线？它们是怎样做到的？让我们 DIY 一个萌萌哒的小车机器人来一探究竟吧！

功能介绍:

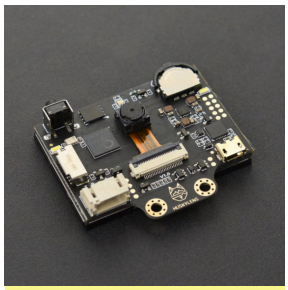
本项目利用 HuskyLens 的巡线功能，让麦昆 plus 按照地面上的线路轨道欢快地蹦跶。

材料清单:



Micro:bit ×1

Maqueen Plus ×1



HUSKYLENS ×1

知识园地:

如果我们要让小车机器人按照地面上的线条移动，就需要一些传感器来识别这些线条，根据传感器的不同，巡线方式也分几种，我们这个项目是使用视觉传感器——HUSKYLENS 的巡线功能来实现巡线效果的。

一、什么是巡线?

巡线（同“**循线**”），就是通过传感器探测地面色调迥异的两种色彩从而获得引导线位置，修正机器人运动路径的一种技术。功能完整的巡线机器人是以移动机器人作为载体，以可见光摄像机、红外热成像仪、其它检测仪器作为载荷系统，以机器视觉—

电磁场—GPS——GIS 的多场信息融合作为机器人自主移动与自主巡检的导航系统，以嵌入式计算机作为控制系统的软硬件开发平台。所以巡线是一个相当复杂的过程，快速、精准的巡线是我们共同的追求。

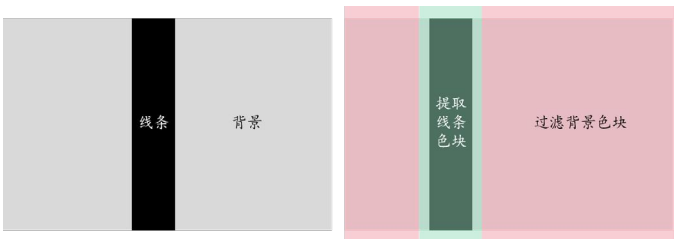
二、两种常用巡线方式的对比

种类 / 对比项	红外巡线传感器	视觉传感器
成本	低	高
视野	传感器需贴近地面使用，视野范围小	视野范围大，能根据线条变化提前调整运动状态
环境适应	改变使用环境时，需要调节传感器灵敏度，且调节过程繁琐	改变使用环境时，只需要重新学习线条即可，操作简单
地图适应	一般只适合背景线条颜色分明、黑色与白色线条、实线的简单地图	可适应背景线条颜色分明、多种颜色线条、实线与虚线等复杂条件下的地图

三、HUSKYLENS 传感器 - 巡线功能原理

HUSKYLENS 的巡线功能是基于卡内基梅隆大学的开源项目 pixy 实现的。

pixy 的算法能对图片进行颜色识别，其基本思想是利用颜色空间来除去所有用户不感兴趣的背景，提取出前景（如线条）。



这样 HUSKYLENS 在学习线条的颜色后就会识别出拍摄视野范围内的该颜色线条。

四、HUSKYLENS 传感器 - 巡线功能演示

学习物体:

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。本项目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。

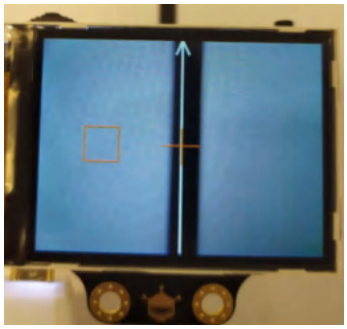
5、向左拨动 “功能按键”，选中 “保存并返回 “，短按” 功能按键 “，屏幕提示” 是否保存参数？“，默认选择” 确认 “，此时短按” 功能按键 “，即可保存参数，并自动返回到巡线模式。

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：

1、学习线条时，尽量把 HuskyLens 的位置调节到与线条平行。

2、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。

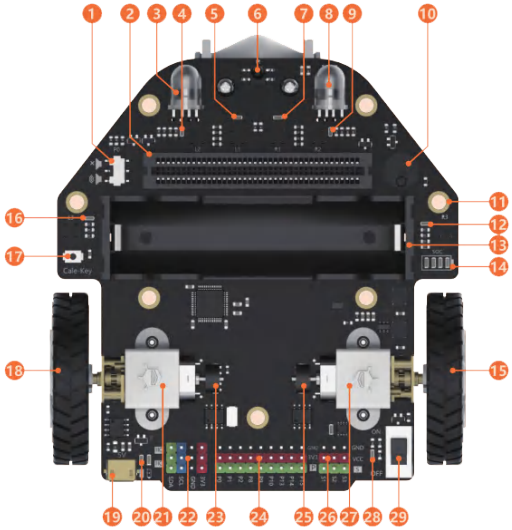
3、HuskyLens 可以根据线条的颜色不同，进行多个线条的学习与巡线，但是线条必须是与背景有明显色差的单色线条。大多数情况下，巡线用的线条，只有一种颜色。为了保证稳定性，建议只对一种颜色的线条进行巡线。

4、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。

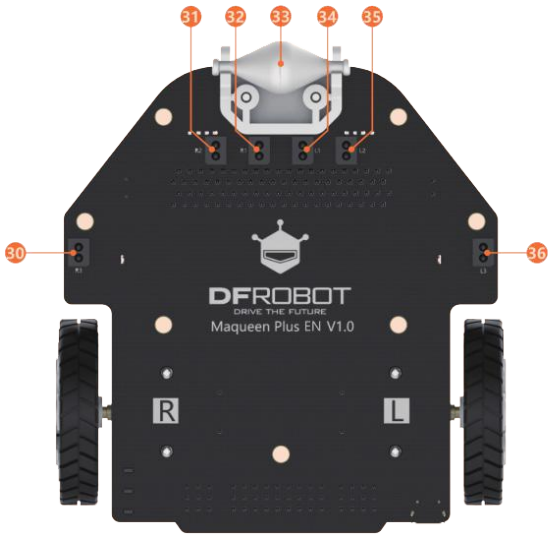
五、Maqueen Plus

麦昆 plus 是 DFRobot 面向教育用户推出的一款产品，拥有丰富的功能和扩展，具有以下特色功能：

- 兼容 microbit 和掌控两种主控板，一键切换
- 18650 大容量锂电池供电，板载充电电路，续航时间长
- 板载编码器车速传感器，可实时获取当前车况、车速，例如可获取麦昆 plus 当前的车速，是停止还是前进、后退，左转还是右转，转弯大小
- 可切换为 PID 电机车速控制，切换为 PID 后，电机车速的误差将变的非常小
- 8 路 GPIO 扩展接口，3 路 I2C 接口，3 路舵机专用接口，扩展口数量多
- 6 路巡线传感器，支持巡线传感器一键校准，支持模拟量输出，可实现更复杂的巡线逻辑，适用于拓展训练及比赛场景
- 支持 python 编程
- 配置金属扩展顶板、可调角度支架，结构扩展能力强



1 蜂鸣器开关
2 micro:bit 插槽
3 RGB-LED-L
4 L2 指示灯
5 L1 指示灯
6 红外接收器
7 R1 指示灯
8 RGB-LED-R
9 R2 指示灯
10 蜂鸣器
11 M3 安装孔
12 R3 指示灯
13 电池盒
14 电量指示灯
15 右轮
16 L3 指示灯
17 巡线校准键
18 左轮
19 充电口
20 充电指示灯
21 左侧电机
22 I2C扩展口
23 左侧编码器
24 GPIO 口
25 右侧编码器
26 舵机口
27 右侧电机
28 供电电源指示灯
29 供电电源开关



30 R3 巡线传感器
31 R2 巡线传感器
32 R1 巡线传感器
33 支撑轮
34 L1 巡线传感器
35 L2 巡线传感器
36 L3 巡线传感器

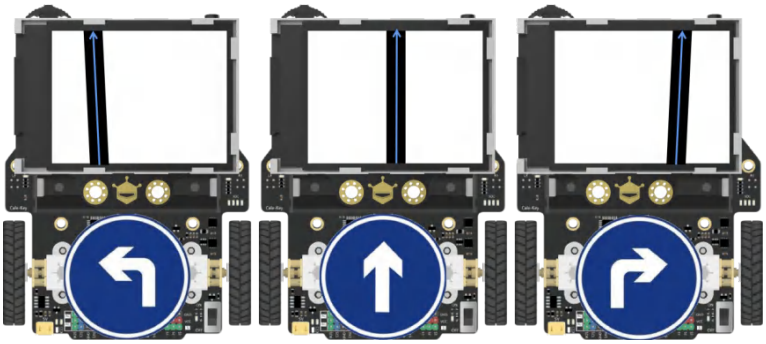
六、HUSKYLENS 巡线实现逻辑

如何让麦昆 plus 在白底黑线的巡线地图上循着黑线运动呢？其实我们只需要知道麦昆 plus 相对黑线是什么位置，分下面三种情况：

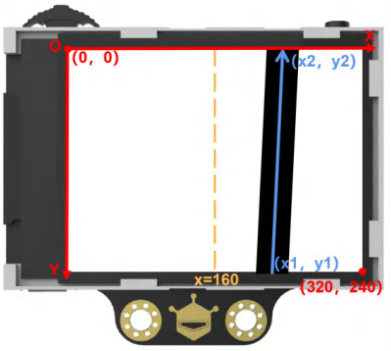
1、当麦昆 plus 在黑线的偏右位置时，控制麦昆 plus 向左转弯；

2、当麦昆 plus 与黑线相对居中时，控制麦昆 plus 直行；

3、当麦昆 plus 在黑线的偏左位置时，控制麦昆 plus 向右转弯。



具体应该怎么实现呢？我们将 HuskyLens 在巡线过程中屏幕上显示的信息剥离出来，抽象成下图的几何数学模型。



HuskyLens 屏幕的分辨率是 320×240，屏幕左上角的 O 点为屏幕的坐标原点(0, 0)，水平向右方向为 X 轴正方向，竖直向下方向为 Y 轴正方向，因此屏幕右下角的坐标为(320, 240)。上图中橙色虚线为屏幕的中轴线，这条线的横坐标值为 160。上图屏幕中黑色的线，是 HuskyLens 摄像头“看到”的巡线地图线条，蓝色箭头为 HuskyLens 计算出来的线条方向，蓝色箭头的起点坐标为（x1, y1），终点坐标为（x2, y2）。

综上，所以我们只需要判断蓝色箭头的起点相对中轴线的位置就能实现巡线了。

项目实践：

我们将按照巡线逻辑实现项目，分若干步不断优化巡线效果，使麦昆 plus 能既快又稳地通过地图。首先我们会学习使用 HuskyLens 的巡线功能，读取线条的横坐标数据，编写简单（二区间一线）的调整运动状态程序满足巡线要求，然后根据调试效果改进我们的项目程序。

任务一：开始巡线

将 HuskyLens 的屏幕横轴分为两个区间（向左向右）和一条线（中轴线）实现巡线效果。

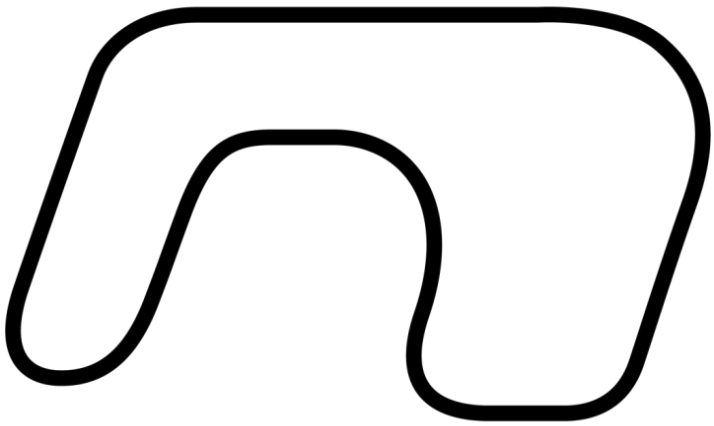
任务二：巡线优化 1

在任务一的基础上，增加直线运动调节区间，加快巡线速度，改善任务一巡线结果中出现的问题。

任务三：巡线优化 2

在任务二的基础上，增加更多运动调节区间，加快巡线速度，继续改善前次巡线结果中出现的问题。

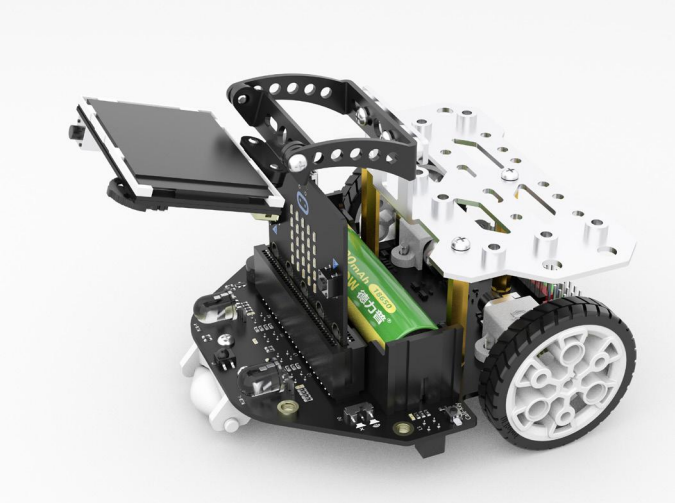
巡线地图：



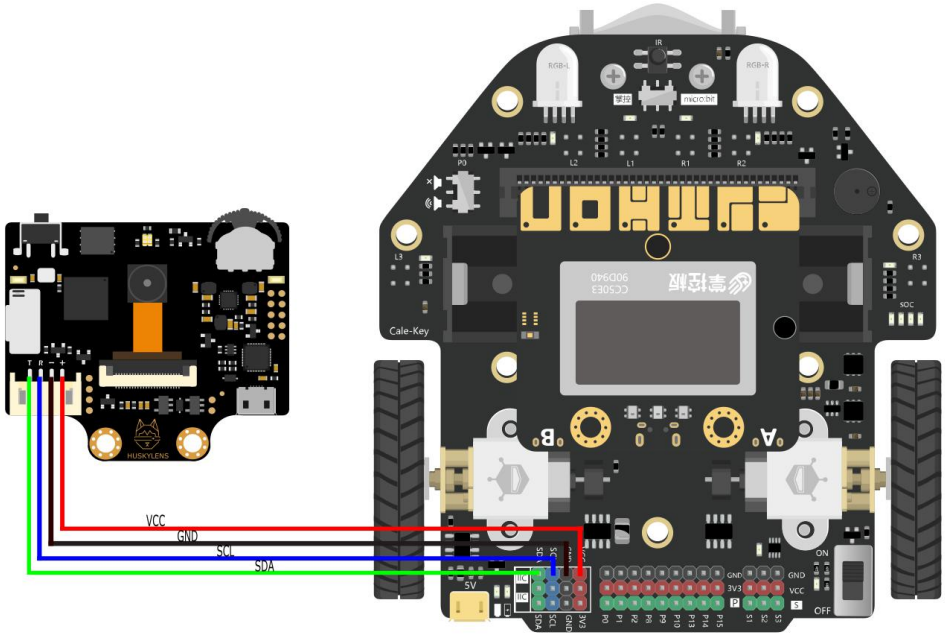
任务一: 开始巡线

1 结构搭建及硬件连接

使用螺丝固定 HuskyLens 与麦昆 plus，需要注意的是，为了巡线，我们需要将摄像头斜向下调节，这样能够看到离麦昆麦昆 plus 更近距离的黑线，巡线效果更佳。

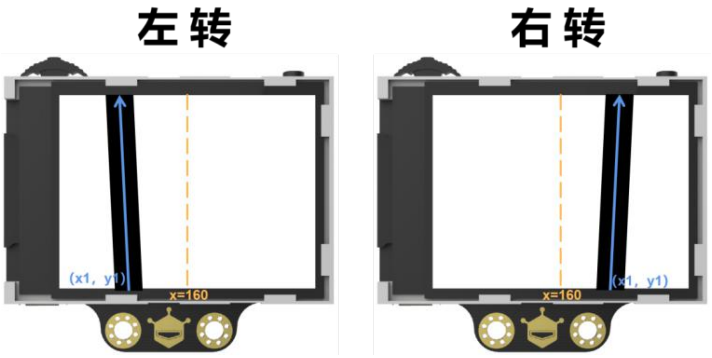


HuskyLens 与麦昆 plus 通过 I2C 通信连线图如下:

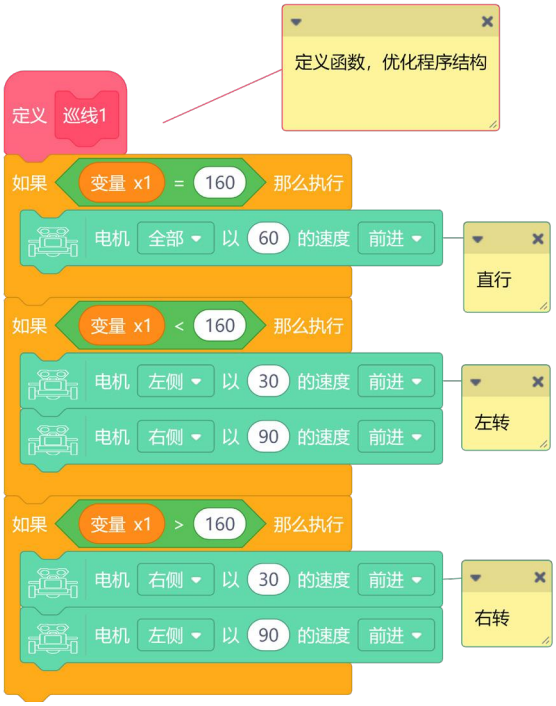


2 程序设计

当 HuskyLens 检测到黑线在屏幕的左边时，即蓝色箭头的起点横坐标值 x1<160，控制麦昆 plus 左转；当黑线在屏幕的右边时，此时 x1>160，控制麦昆 plus 右转；当黑线在屏幕的中间时，此时 x1=160，控制麦昆 plus 直行。



3 程序示例



4 运行效果

麦昆 plus 能完成基本的巡线任务，但同时也暴露如下几个问题：

1、麦昆 plus 在前进中明显左右晃动，速度变化不连贯，不能稳定直行运动；

2、速度不能设置太快，速度太快在转弯处很容易脱线；

3、不同的转弯角度，需要的转弯速度也不一样，当巡线地图中有好几种转弯角度时很容易脱线。

（* 如果转弯过程中麦昆 plus 脱离轨道，就需要通过修改左右侧电机的速度，不断调试。）

任务二：巡线优化 1

1 结构搭建及硬件连接

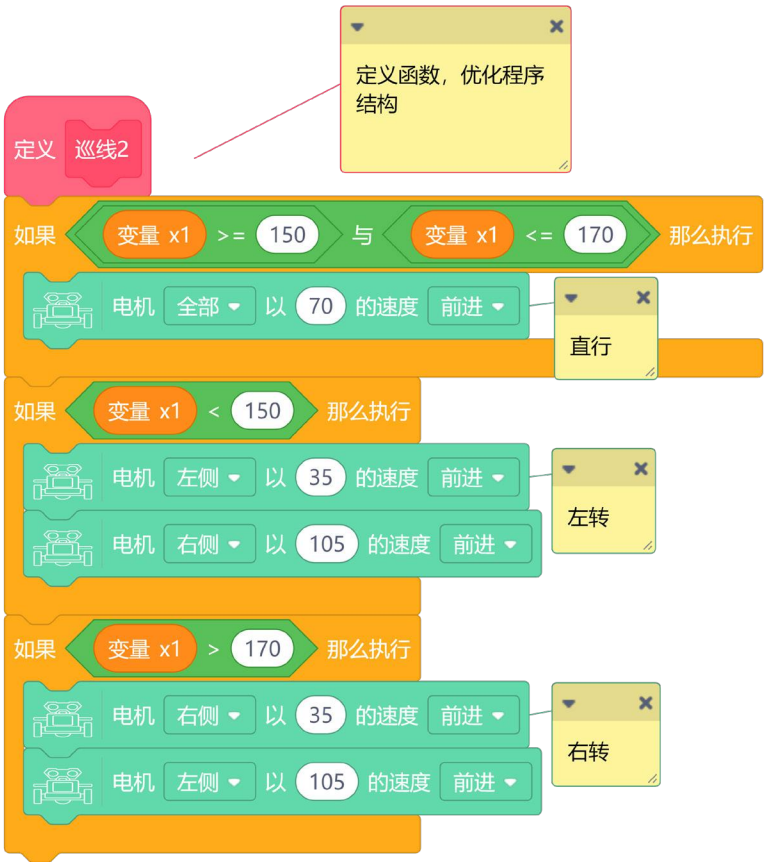
同任务一

2 程序设计

任务一中麦昆 plus 在前进中左右晃动，不能稳定直行运动，这是为什么呢？因为麦昆 plus 直行的横轴区间只是一条线，而运动过程中带有惯性，所以在 x=160 处直行是很难实现的。

优化思路是将直行的运动区间扩大，如下图所示，我们将区间 [150，170] 设置为麦昆 plus 直行区间，当起点坐标值 x1 在这个区间内时，控制麦昆 plus 直行；当 x1<150 时，控制麦昆 plus 左转；当 x1>170 时，控制麦昆 plus 右转。

3 程序示例



4 运行效果

通过调整，麦昆 plus 的巡线速度快了一些，在直线轨道上，速度变化也顺滑了一些，但在不同的转弯角度上仍然容易脱线。

任务三：巡线优化 2

1 结构搭建及硬件连接

同任务一

2 程序设计

前面任务中之所以出现转弯脱线的问题，是因为同一种转弯速度并不能适应所有

的转弯角度，一旦设置的转弯速度过快或过慢，都会导致脱线。按照我们先前的优化思路，既然可以将屏幕分成 3 个运动调节区间，为什么不继续分为 5 个呢？黑线位置越趋近两边，转弯速度越大；黑线位置越趋近于中轴线，转弯速度越小。

或者更进一步,直接分为 7 个运动调节区间呢？

3 程序示例


定义函数，优化程序结构
定义 巡线3
如果 变量 x1 >= 150 与 变量 x1 <= 170 那么执行
电机 全部 以 80 的速度 前进
直行
如果 变量 x1 >= 130 与 变量 x1 < 150 那么执行
电机 左侧 以 60 的速度 前进
电机 右侧 以 100 的速度 前进
左小转
如果 变量 x1 >= 110 与 变量 x1 < 130 那么执行
电机 左侧 以 50 的速度 前进
电机 右侧 以 110 的速度 前进
左中转
如果 变量 x1 < 110 那么执行
电机 左侧 以 40 的速度 前进
电机 右侧 以 120 的速度 前进
左大转
如果 变量 x1 > 170 与 变量 x1 <= 190 那么执行
电机 左侧 以 100 的速度 前进
电机 右侧 以 60 的速度 前进
右小转
如果 变量 x1 > 190 与 变量 x1 <= 210 那么执行
电机 左侧 以 110 的速度 前进
电机 右侧 以 50 的速度 前进
右中转
如果 变量 x1 > 210 那么执行
电机 右侧 以 40 的速度 前进
电机 左侧 以 120 的速度 前进
右大转

4 运行效果

麦昆 plus 的巡线速度可以更快了，不管是转弯还是直行，速度变化都顺滑很多了，脱线的次数也减少了。

项目小结：

项目回顾：

本项目中我们通过不断优化巡线算法，使巡线的效果越来越快速平稳，在实际生产中机器人需要在更复杂的环境中巡线，这就可能需要视觉巡线与普通巡线双管齐下，选择线路的最优解。

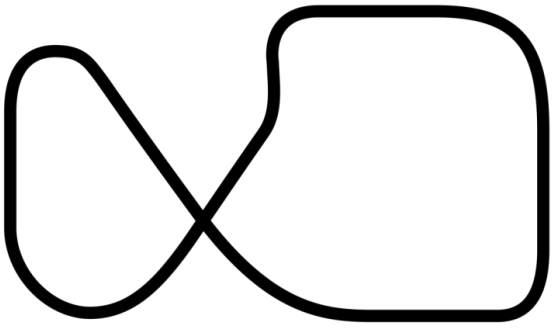
知识点回顾：

1、学习了巡线的主要实现思路

2、巡线过程中的算法优化

项目拓展：

在完成了轨道不交叉巡线后，可能需要处理如下图所示的轨道交叉巡线，可不可以用前面的标签识别让麦昆 plus 在交叉点选择正确的路径呢？



拓展知识：

麦昆巡线时的巡线区间可以分为 2 个、3 个、5 个、7 个，那是不是还能继续细分呢？9 个？ 11 个？……直到无限个。按照本项目的实际效果，调速区间越细分，巡线效果越好，但是这样写程序会越来越长，有什么办法呢？

PID 调速算法可以帮助我们解决这个问题，PID 全称比例、积分、微分控制，它是一种闭环控制系统，闭环控制是根据控制对象输出的反馈来进行校正的控制方式，它能够根据测量出的实际值与计划值之间的误差 Error，按一定的标准来进行纠正。

有兴趣的同学可以在网上搜索相关信息，继续优化我们的巡线效果。

扫码进入**创客世界**

DFRobot创客社区