2018 年广东省大学生电子设计竞赛

设计报告

基于深度学习的办公室智能分类垃圾桶
Intelligent sorting office garbage bin based on deep learning

摘要：本作品设计的是应用于办公室的智能分类垃圾桶，基于人工智能深度学习模型和运用树莓派、STM32 嵌入式模块，实现自动检测开盖，垃圾种类识别、自动分类处理和容量实时监测等主要功能，配合 LCD 显示屏动态显示结果，推广使用能够有效解决国内垃圾分类现状，提高资源化利用率。

Abstract： This work is designed for intelligent sorting bins in offices. Based on the artificial intelligence depth learning model and using the raspberry pie and STM32 embedded module, it realizes the main functions of automatic detection and uncovering, garbage category identification, automatic classifying and processing, and real-time capacity monitoring, and so on. Dynamic display results with LCD display. Popularization and application can effectively solve the domestic garbage classification status and improve the utilization rate of resources.

原 创 性 声 明

本团队郑重声明：所呈交的作品及论文，是本团队在指导老师的指导下，独立进行研究所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的科研成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本声明的法律责任由本人承担。

作者签名：_____ 日 期：_____

目　录

一、 查新说明

查新作品名称	基于深度学习的办公室智能分类垃圾桶

一.查新目的

申报 2018 年广东省大学生电子设计竞赛——“人工智能”专题竞赛

二.查新作品的技术要点

据中国城市环境卫生协会的统计数据显示，全国城市生活垃圾年产量巨大，而我国垃圾的分类收集工作不到位导致了垃圾资源化利用率低下，因此，我们把当下社会热点结合人工智能，研制智能分类垃圾桶。

人工智能作为当下科技热点，应用于机器人、语言识别、图像识别等多个领域。将人工智能应用于垃圾分类，需要解决如何实现自动检测、如何准确识别多类别的生活垃圾、如何设计合理的机械结构、如何自动分类处理等技术问题，达到研制出一款能够自动检测、准确识别垃圾、自动分类处理的智能分类垃圾桶的目的

三.查新点

本作品的主要创新点包括：

1.利用简单的材料自行设计机械结构，能够实现传动功能
2.能够准确识别多种类的生活垃圾
3.能够自动进行垃圾分类处理
4.能够实时垃圾桶容量检测

四.文献检索范围及检索策略

（一）文献检索范围：

查新使用的数据库：

1.中国知网中国期刊全文数据库	1994——2018.8
2.维普中文科技期刊数据库	1989——2018.8
3.万方数字化期刊全文数据库	1983——2018.8
4.中国科技成果数据库	1994——2018.8
5.中国专利数据库	1985——2018.8

6.百度、Google 等搜索引擎

（二）检索策略

检索词：人工智能、深度学习、机器学习、垃圾桶、垃圾分类

检索式：（1）（机器学习 or 深度学习）and （垃圾桶 or 垃圾分类）

（2）人工智能 and（垃圾分类 or 垃圾桶）

五.检索结果

依据上述检索范围和检索式，共检索到相关文献 80 多篇，现将与该查新作品密切相关的 6 篇文献，摘录如下：

（1）【题名】一种智能分类垃圾桶系统的设计

【作者】熊建桥 陆卫 季小涵 万锦晓 张志刚 付杰

【机构】南京工程学院机械工程学院

【刊名】机电产品开发与创新 2016 年 05 期

【文摘】论文设计了一种智能分类垃圾桶,以实现垃圾自动分类,简化垃圾分类流程,有效回收利用资源。智能分类垃圾桶由机械结构、硬件电路、软件程序等部

分组成。样机采用皮带传动机构、剪式机构、丝杠螺母机构等机械结构,融合红外感应传感器、金属传感器等,运用典型 STC89C51 控制芯片,初步实现了垃圾桶自动开合、检测分类、压缩储存等主要功能。

（2）【题名】智能分类垃圾桶的设计与实现

【作者】陈子恒 周志成 王梦帅

【机构】东南大学成贤学院电子与计算机工程系

【刊名】机电信息 2017 年 09 期

【文摘】设计了一种能够自动分类金属与非金属,能在垃圾桶已满时不打开顶盖,同时进行物联网通知的智能分类垃圾桶,介绍了其软硬件架构,并对实际案例应用进行了分析。

（3）【题名】新型智能垃圾桶的设计方案

【作者】周慧珺 许锦标

【机构】广东工业大学自动化学院 广东工业大学自动化学院

【刊名】广东工业大学学报 2017 年 09 期

【文摘】提出了一种新型智能垃圾桶的设计方案.采用基于单片机的智能化红外感应控制系统、无线通信系统,实现自动开关桶盖并调节开盖角度、遥控垃圾桶走动、智能封袋的功能.与传统智能垃圾桶相比较,功能性更强,智能化程度更高.

（4）【题名】太阳能智能垃圾桶的设计

【作者】戴礼骁 蒋陈苗 薛兵兵 林云宏 吴陈燕

【机构】台州职业技术学院电子电气工程系

【刊名】《电子技术（上海）》2013 年第 12 期

【文摘】针对现有垃圾桶的露天投放、不能有效地分类回收、易产生臭味等问题,设计了一款太阳能智能垃圾桶。该垃圾桶具有较好的外观,可以有效分类回收,自动翻盖,利用太阳能供电实现照明、语音提示、除臭等功能。智能垃圾桶的设计力求简洁、环保、和谐。

（5）【题名】公共场所智能垃圾桶系统的设计

【作者】鞠海翔 樊东燕

【机构】江苏省南京市第九中学 山西大学商务学院

【刊名】《山西电子技术》2017 年第 6 期

【文摘】本设计针对公共场所垃圾桶影响周围环境的问题而设计。系统应用红外传感器、人体检测装置、单片机、语音装置、舵机驱动设计实现垃圾桶的智能化,使得垃圾桶能通过感应实现自动打开与关闭,以及夜晚的照明和语音提示功能,提升公共场所的环境质量,实现洁净、环保的目的。

（6）【题名】城市中心垃圾分类智能装置的设计

【作者】朱颖文 陈思宇 张驰 熊俊杰

【机构】河海大学能源与电气学院

【刊名】科学与信息化

【文摘】本发明设计了一种在城市中心可自动分类的垃圾箱,该分类箱可对果皮、纸张、金属等不同材质的垃圾进行自动识别,分类,根据垃圾桶的分类会进行语音播报提醒行人该垃圾所属类别.整个系统可由太阳能直接供电.

六.查新结论

针对本作品内容，在上述检索范围内，检索了国内数据库 6 个及相关网站，

检索到与本作品相关文献 80 多篇，其中密切相关文献 6 篇，通过分析对比，得出查新结论如下：

文献（1）利用红外感应传感器、金属传感器等传感器进行自动检测，结合皮带传动结构，但本作品是利用接近光电开关进行检测，利用管道下落结构进行分类处理

文献（2）利用的是多种传感器，能够自动识别和分类金属与非金属材料，但本作品是基于人工智能深度学习框架，对垃圾进行智能识别和处理

文献（4）（5）设计的智能垃圾桶都具有语言提示功能，而本作品不涉及语言模块

文献（6）设计方案将垃圾分为了果皮、金属、纸张、玻璃及塑料等 6 类，运用深度神经网络进行图像识别，而本作品与文献的不同点在于它依照社会普遍的分类规则，将垃圾分为了可回收、不可回收、有毒有害以及其他 4 个类别，应用的是 TensorFlow 卷积神经网络进行深度学习，并结合 OpenCV 进行拍照处理。

综上所述，在国内外公开报道文献中，有多篇关于智能分类垃圾桶的论文文献，但与本作品密切相关的论文和专利数量不多，并没有完全雷同的文献。

二、 系统方案

本系统主要由 stm32 模块、raspiberry 模块、红外开关传感器模块、电源模块、机械传动模块组成。下面分别论证这几个模块的选择。

1. 单片机模块的论证与选择

方案一：采用 STC 系列单片机 STC2C5A60S2

STC12C5A60S2 系列单片机是高速/低功耗/超强抗干扰 的新一代 8051 单片机，指令代码完全兼容传统 8051,但速度快 8-12 倍。内部集成 MAX810 专用复位电路,2 路 PWM,8 路高速 10 位 A/D 转换(250K/S),针对电机控制，强干扰场合。其片内集成了 60kB 程序 Flash，2 通道 PWM、16 位定时器等资源，操作也较为简单，具有在系统调试功能（ISD），开发环境非常容易搭建。但由于设计中要求单片机至少 3 路 PWM 输出，并且要求能控制 LCD 显示图片以及动画等内容，对引脚数量以及功能需求较大，51 单片机无法满足要求，故不选择此方案。

方案二：采用以 STM32F03RCT6 芯片为核心的 STM32 单片机

该 STM32 单片机拥有的资源包括：48KB SRAM、256KB FLASH、2 个基本定时器、4 个通用定时器、2 个高级定时器、2 个 DMA 控制器（共 12 个通道）、3 个 SPI、2 个 IIC、5 个串口、1 个 USB、1 个 CAN、3 个 12 位 ADC、1 个 12 位 DAC、1 个 SDIO 接口及 51 个通用 IO 口。利用该单片机模块，可以进行 LCD 显示，且仍有资源可以进行传感器模块的控制，并能与树莓派进行通信。但该单片机模块相对与 51 单片机来说成本略高。

综合以上两种方案，由于方案二更符合设计需求，故选择方案二。

2. 嵌入式模块的论证与选择

在设计要求中，需要嵌入式模块能运行神经网络模型，进行图像识别以及分类。树莓派像一部微型电脑，能预装 linux 系统，支持无线上网和有线上网，连接显示器等功能，且用有个４个ＵＳＢ口和４０个ＧＰＩＯ口。由于深度学习神经网络模型平台以及环境要求，其他的嵌入式模块并不能很好的完成功能需求。而在树莓派上能更简易的进行深度学习环境的搭建以及模型迁移，故选择树莓派更加合理。

3. 红外感应开关传感器的论证与选择

方案一：选用人体红外感应开关

人体红外感应开关，当有人进入其感应范围内则输出高电平，人离开感应范围后则恢复低电平，且该模块有两种触发方式，分别是可重复触发方式和不可重复触发方式。在实际测试中，人体红外感应开关受外界光线影响较大，且触发信号过程慢。

方案二：选用红外感应光电开关

该光电传感器集发射与接受于一体，发射器对准检测到的目标不断发射红外线光束，接收器把检测物反射回来的光束转换为电流输出给后面的集成电路，经集成电路处理后再放大输出，且调节器检测距离可调。在实际测试中，该传感器反应灵敏，受环境干扰小，但覆盖范围小。

综合以上两种方案，方案二反应灵敏，且受环境干扰小，更加符合设计需求，故选择方案二。

4. 机械传动模块的论证与选择

方案一：选用传送带结构

该结构利用工作构件的旋转运动或往复运动，或利用介质在管道中的流动使物料向前输送，能够通过传送带轻松传送物体。但其要求有足够的刚度来支承物料，并且其需要的空间范围大。

方案二：选用管道下落结构

该结构利用管道左右角度的摆动，促使物体下落到指定的位置，其所占的空间小，但是其要求有足够的深度使物体下落。

综合以上两种方案，由于受限于垃圾桶实际大小，空间范围都不能达到理想的大小，故选择方案二。

三、 系统理论分析与计算

1. 系统分析

由于本设计是一个全自动的垃圾分类装置，涵盖垃圾从投入，到识别，再到分类装桶以及最后对桶容量进行检测的全过程。在投入前，需控制光电接近开关的感应距离，以及开盖舵机的旋转角度；而识别过程，需要确定关盖后照明电路的延时时间，确保图像的清晰采集与节省功耗；分类装桶过程，需要根据垃圾的种类将垃圾装入各自分桶中，故需精准控制分类管道控制舵机的旋转角度。故在本设计中对舵机旋转角度的精准控制以及对光电接近开关的灵敏度调校对整个系统的运行至关重要。而本装置的舵机与光电开关调校也离不开对装置结构的考量与计算。

2. 舵机与光电开关的调校与计算

基于以上考虑，本设计在已经完成的装置基础上进行了以下调校与计算，根据正常情况下投掷垃圾的习惯以及手与垃圾桶盖的正常距离，得到了光电开关合理的感应距离，兼顾不误判与不漏判；在分类舵机的调校上，根据物体下落的速度与对应小桶位置，将 pwm 的精度进行细化，并根据管道的位置，将挡板旋转方向进行特定的选择，然后将 pwm 信号以较低精度输出给舵机，占空比从 5%缓慢调至 25%，确定不同垃圾对应舵机的旋转角度；而对于检测容量的光电开关的调校，本设计将小桶的容量进行刻度化，根据光电开关的感应范围调整光电开关的放置角度与感应距离。

四、 功能与指标

1. 红外感应自动检测开盖
2. 投放垃圾多种类识别
3. 垃圾自动分类处理
4. 垃圾桶实时容量检测
5. 按键复位功能

五、 实现原理

1.功能实现流程

本系统设计的整个功能流程为自动感应开盖，垃圾种类检测，垃圾分类投掷，垃圾桶容量检测，各个阶段对应实现原理如下：

（1）自动感应开盖：通过光电接近开关检测到物体返回信号给 stm32，stm32 检测到后控制舵机从左极限选择到右极限，使得桶盖得以 180 度的旋转，进而实现

自动感应完全开盖，以便垃圾的投掷；
（2）垃圾种类检测：垃圾落入桶后，由挡板挡住落在第一层结构上，之后桶盖自动关闭，照明电路由 stm32 控制的继电器模块连接通路，并且 stm32 向树莓派发送识别开始信号，树莓派通过摄像头拍取一帧图像，通过已经训练好的深度学习模型对垃圾种类进行识别。
（3）垃圾分类投掷：树莓派通过串口通信向 stm32 发送不同种类垃圾所对应的编码，此处为将不同种类的垃圾区分开并且准确传送给 stm32，本设计将不同种类的垃圾进行特征编码，简单易懂且方便调试，并制定了一套特殊的通信协议，让树莓派与 stm32 通信流畅，不受其他不可控因素影响，待 stm32 接收到信号后，根据其所属垃圾类别，发送给舵机不同的 pwm 信号，让舵机带着管道旋转，之后控制挡板舵机打开挡板，垃圾自由下落，在重力作用与管道引导下滑落至对应分桶。
（4）垃圾桶容量检测：垃圾滑入对应分桶后，相应桶的光电开关开始进行检测，若桶内垃圾达到一定容量，光电开关检测距离适合，光电开关会被触发，发送信号给 stm32，stm32 获得信号后在显示屏显示出是否满的特征图像。

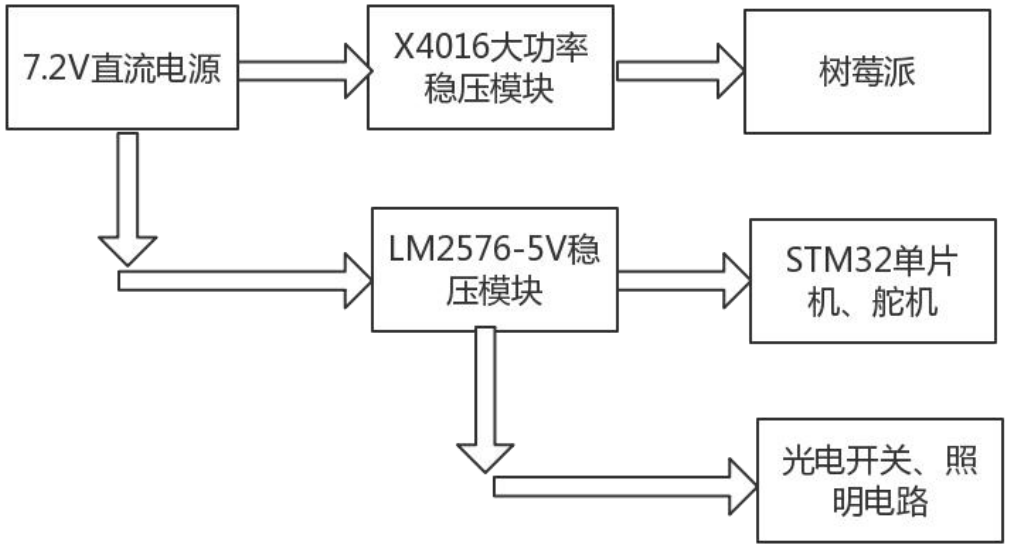
2. 深度学习模型搭建

搭建人工智能深度学习模型，选用谷歌 TensorFlow 人工智能学习架构，根据作品需要搭建卷积神经网络。卷积神经网络是一种前馈神经网络，它的人工神经元可以响应一部分覆盖范围内的周围单元，对于大型图像处理有出色表现。它包括卷积层和池化层。通过卷积层对图片进行图像特征处理，通过池化层对输入的特征图进行压缩，一方面使特征图变小，简化网络计算复杂度；一方面进行特征压缩，以此来提取主要特征。

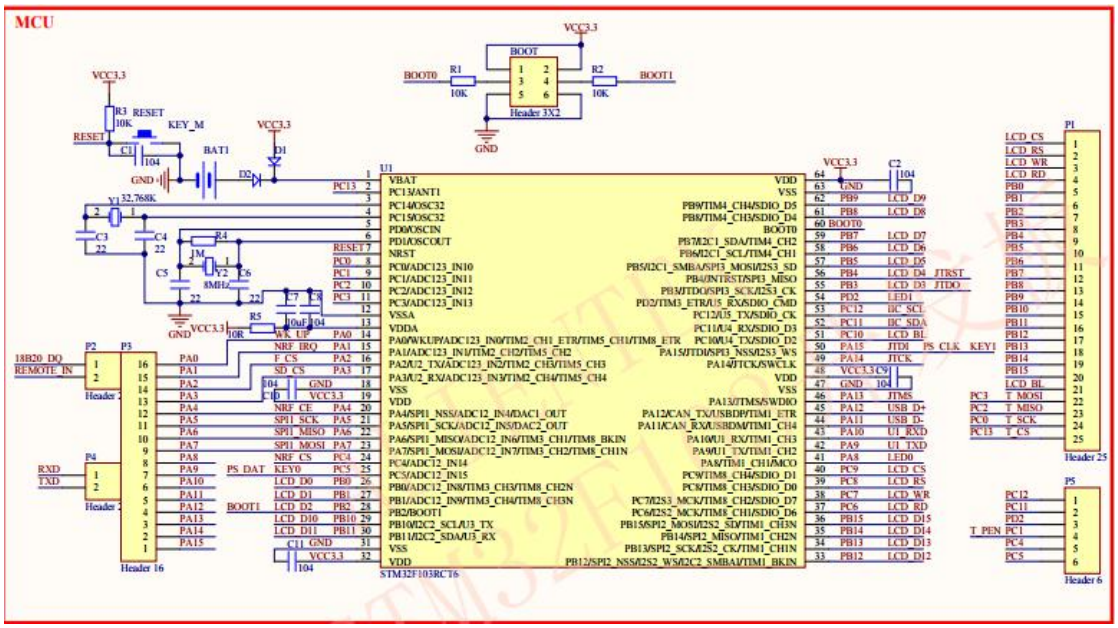
训练出来的适用于本作品的深度学习模型，输入层是 6 个种类的训练数据集，每个类别的数据集大概有 500 张训练图片，隐藏层是 96 个卷积层，而输出层是 6 个种类的标签值。

六、 硬件框图

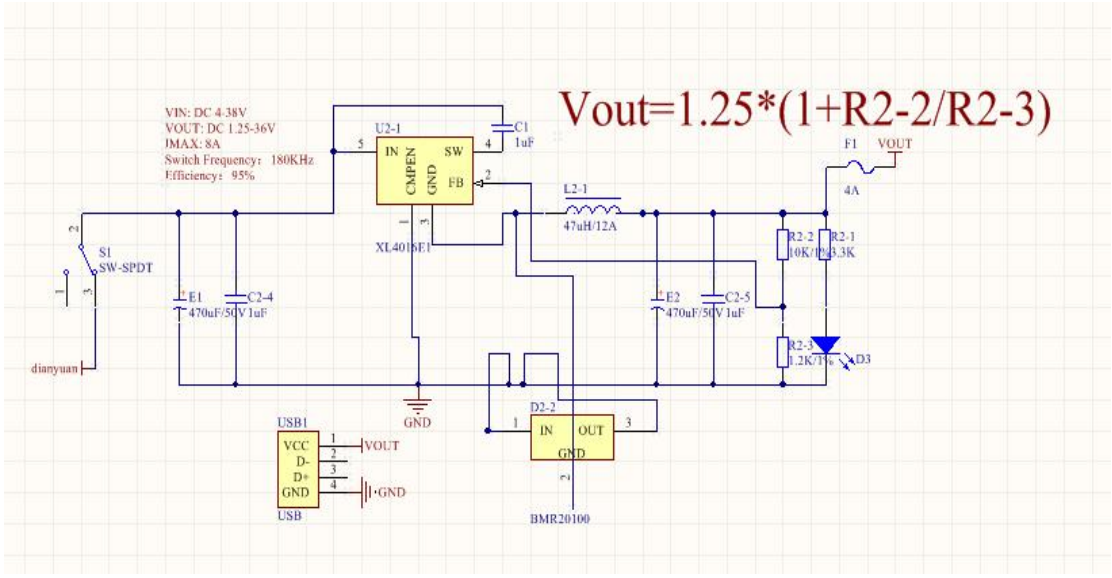
1. 系统总体电路框图



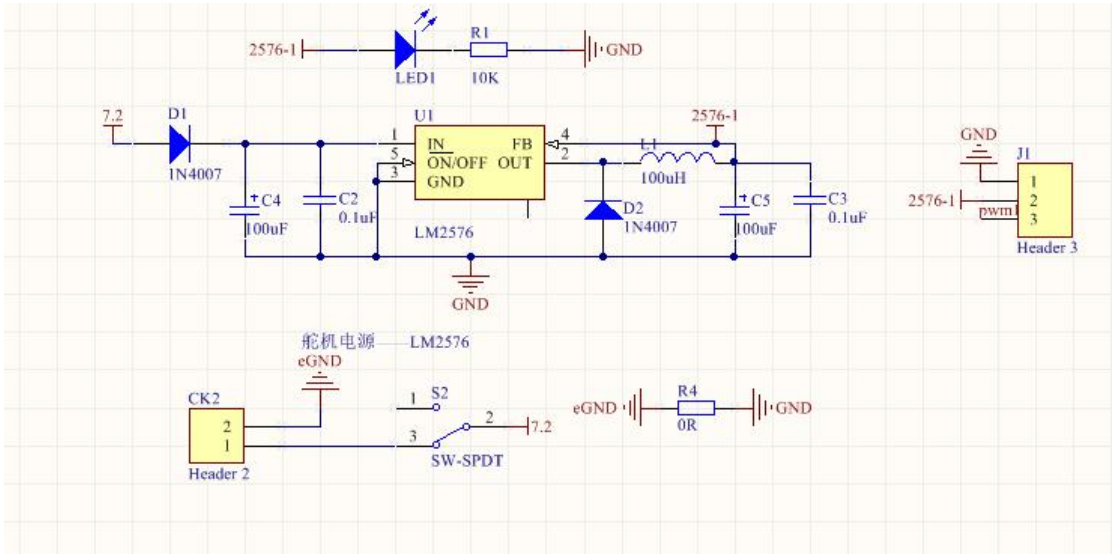
2. STM32 CPU 原理图



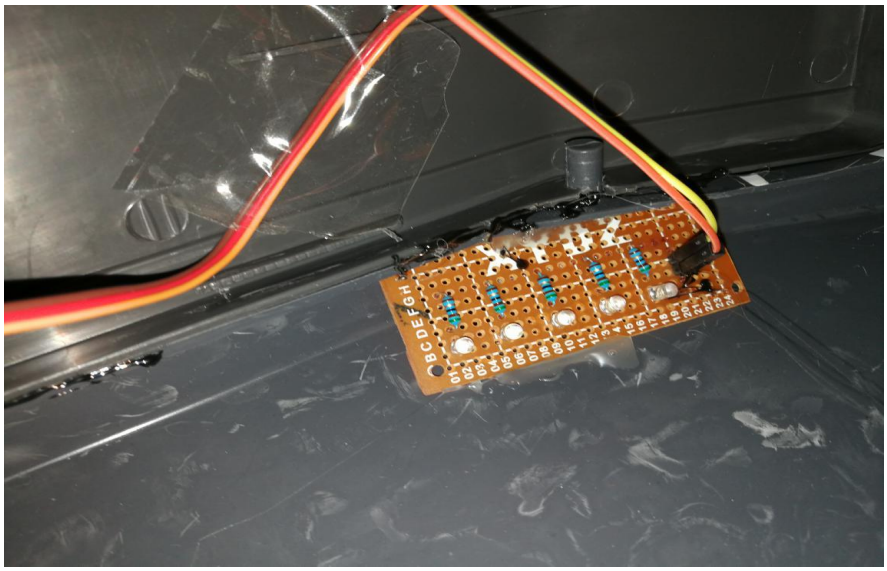
3. X4016 稳压电路原理图



4. LM2576 稳压电路原理图



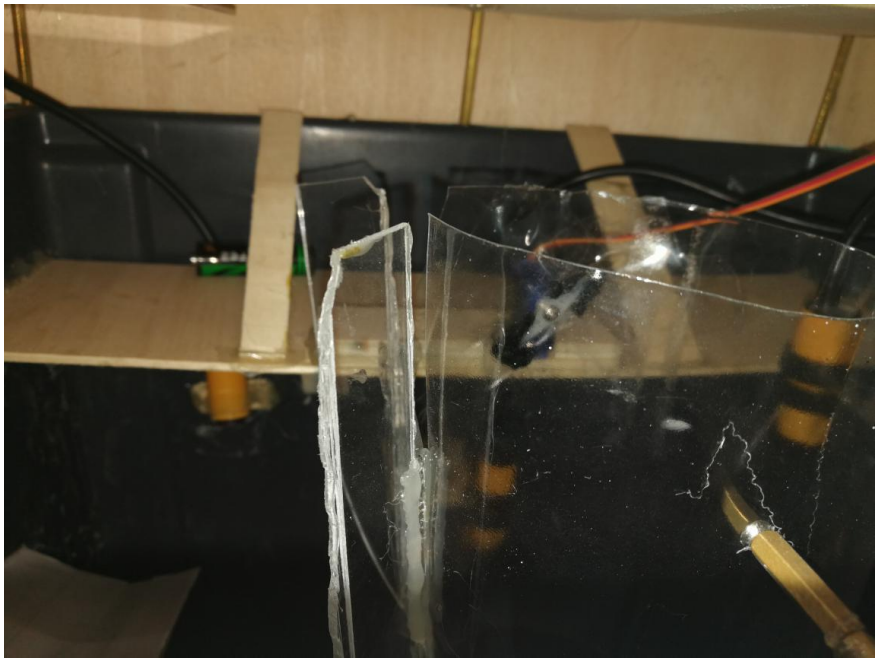
5. 照明模块电路实物图



6. 舵机控制电路结构

（1）舵机控制门转向

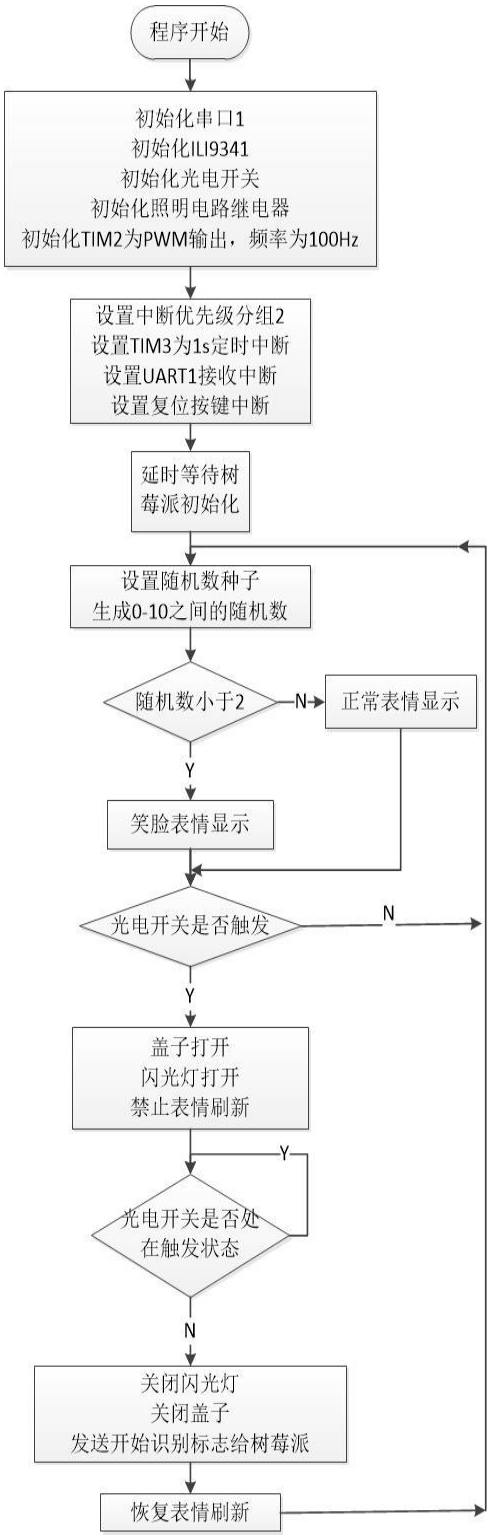
（2）舵机控制滑道方向



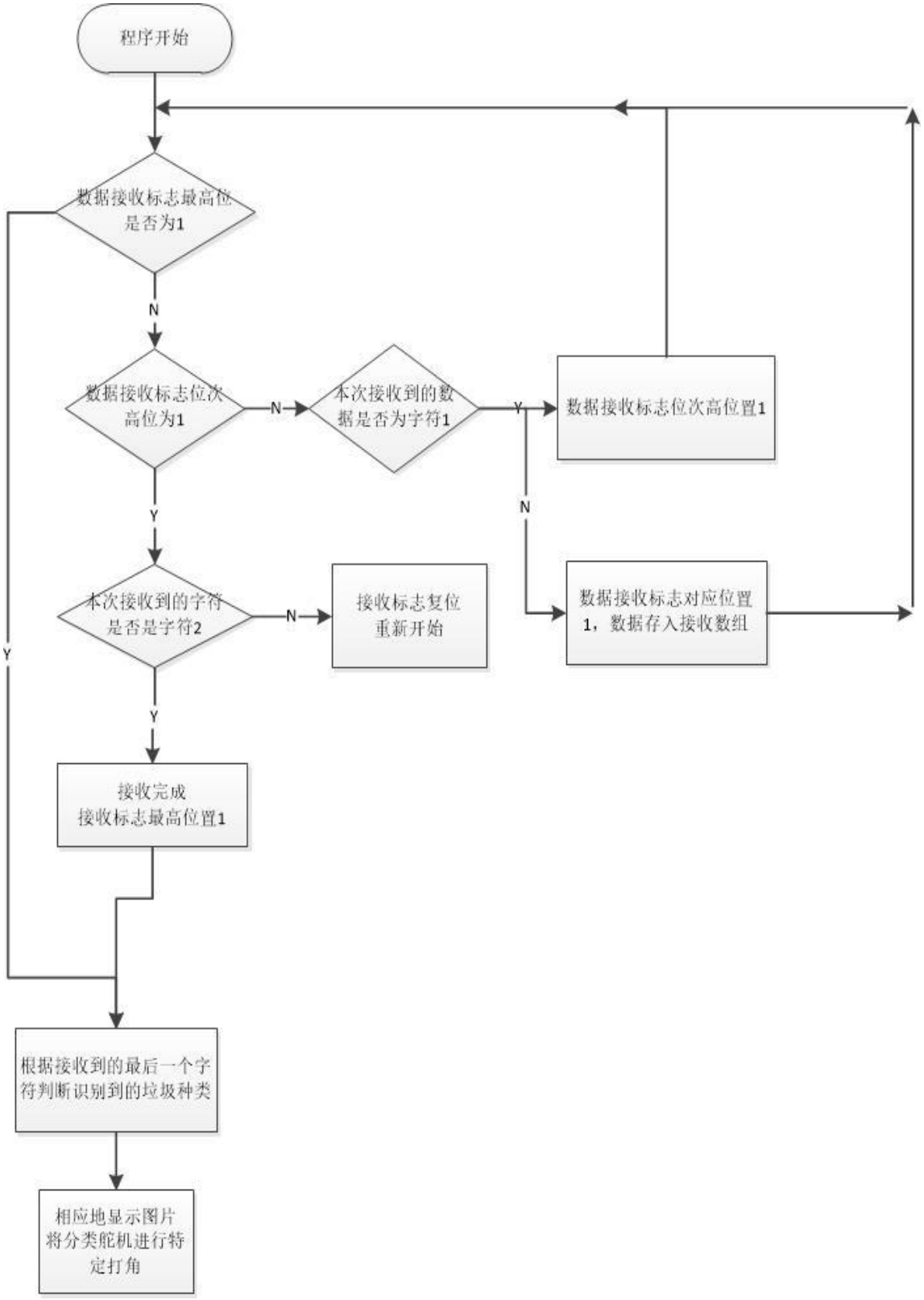
七、 软件流程及外观设计

1. 软件设计

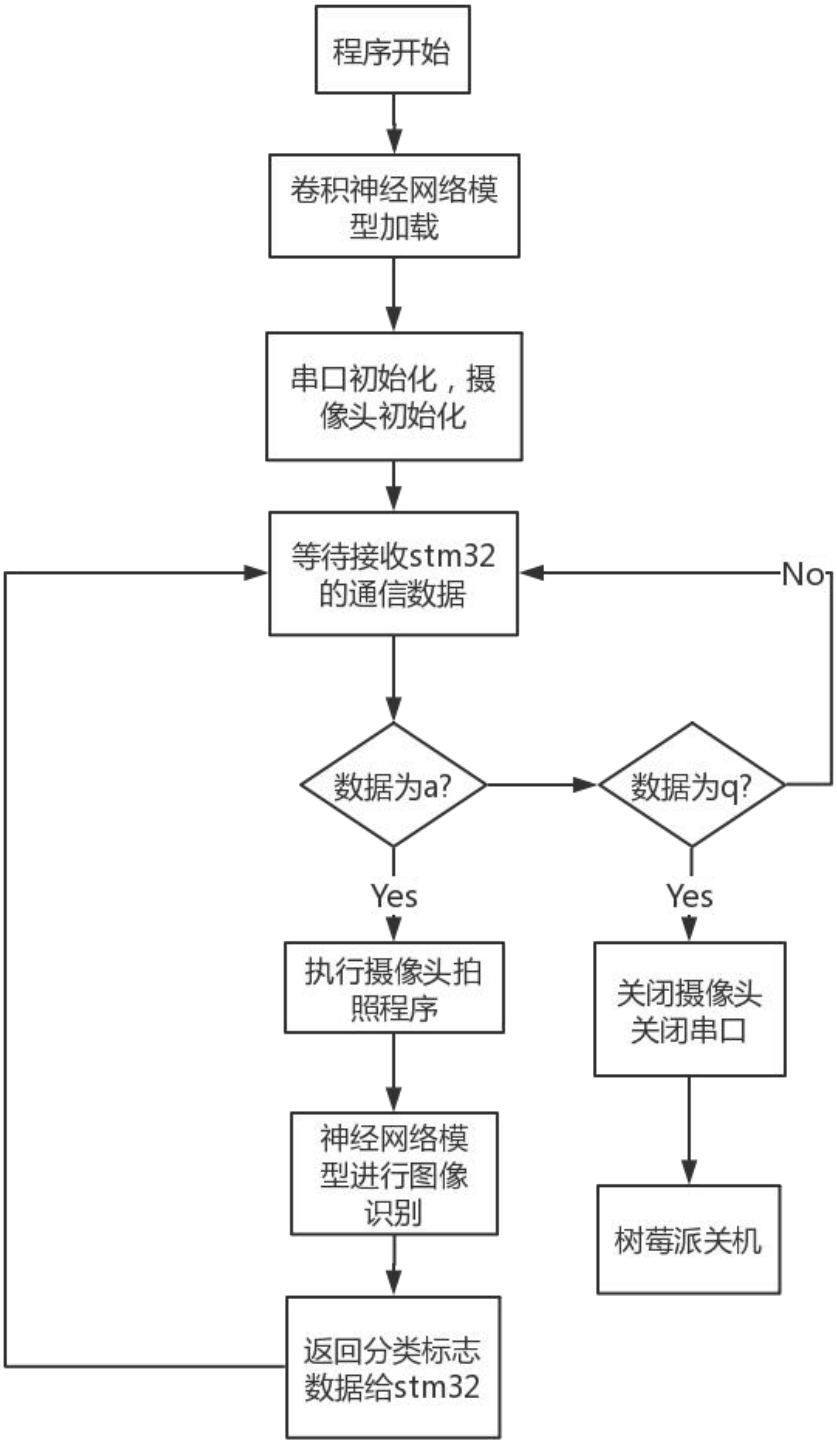
（1）STM32 主程序流程框图



（2）串口接收中断程序流程框图



（3）嵌入式系统树莓派程序框图



2. 外观设计

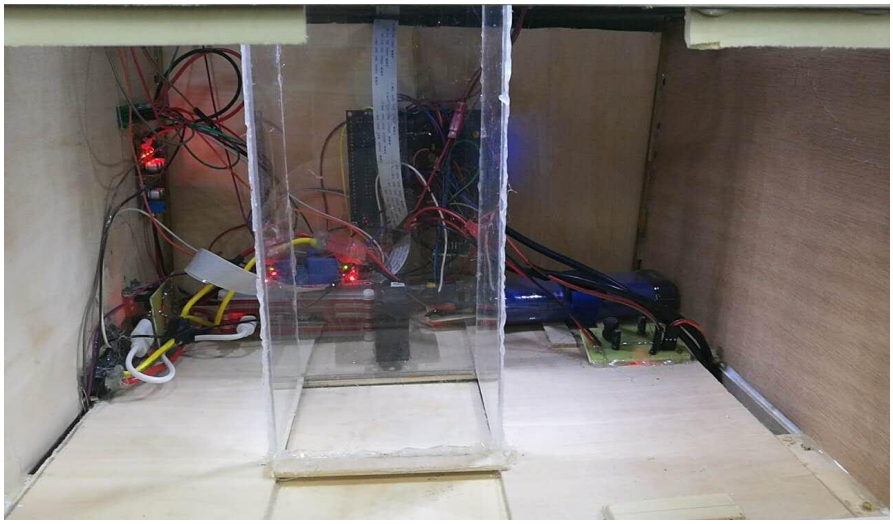
（1）外部结构

本系统外观设计参考《机器人瓦力》中的机器人伊娃造型，整体色调为黑白两色，桶盖部分天然的形状如帽子一般，与下方架高的木箱结构形成本设计的头部造型，而在脸部，显示屏通过相同的颜色融入其中，通过程序设计的表情交互界面，使得呆板的造型具有活力。在桶身下部，本装置通过上色画上了两只向内倾的机器手，结合显示屏的表情动画给人一种乖巧可爱的整体效果。

（2）内部结构

本系统内部结构设计分为三层结构：

第一层为识别层，垃圾进入后停留在此层并进行识别，同时此层是系统电路的主要集合处，集成了 stm32，树莓派及电源等复杂电路；



第二层为分类层，主要由分类管道构成，管道在此层结构能随舵机自由旋转；

第三层为分桶层，存在各类垃圾的容纳桶，同时背后为有一个随合页旋转，由卡扣固定的开盖，方便垃圾桶的取出。

下图为本设计的外观与内部结构简析图：

八、 测试方案与测试结果

1. 测试方案

（1）硬件测试：滑道根据舵机的摆角可以左右旋转，当垃圾掉落时，滑道的角度必须对应桶的方位才能使垃圾掉落到对应的分类桶内。利用 stm32 输入信号，使舵机摆角，丢垃圾进入滑道，看垃圾能否通过滑道掉到对应的垃圾桶上。通过不断测试得到各个分类的摆角的信号值。

（2）硬件软件联合调试：调试时，将 stm32 和树莓派以及各个模块电路连接好，通过总电源开关开始运行程序和结束程序，看完成整个流程下来显示屏显示是否正常，垃圾是否落到了对应的桶上，记录每次的结果，不断调整程序，得到最优解。

2. 测试条件与仪器

（1）测试条件：确认电路无虚焊，电路与原理图相同，信号连接线连接正确且牢固。

（2）测试仪器：数字万用表、SK3303 直流电源、F40 型数字合成函数信号发生器、模拟示波器

3. 测试结果及分析

（1）硬件测试：

不可回收垃圾桶摆角测试

信号值	784	794	799	809	814	819	824	829	834
结果	不能掉入	不能掉入	不能掉入	不能掉入	能掉入	偶尔掉入	不能掉入	不能掉入	不能掉入

可回收垃圾桶摆角测试

信号值	834	844	849	854	859	864	869	874	879

结果	不能掉入	不能掉入	不能掉入	能掉入	不能掉入	不能掉入	不能掉入	不能掉入	不能掉入

有毒有害垃圾桶摆角测试

信号值	889	909	914	919	924	929	934	939	944
结果	不能掉入	不能掉入	不能掉入	不能掉入	不能掉入	偶尔掉入	能掉入	不能掉入	不能掉入

其他垃圾桶摆角测试

信号值	869	874	879	884	889	894	899	904	905
结果	不能掉入	不能掉入	不能掉入	不能掉入	能掉入	偶尔掉入	偶尔掉入	不能掉入	不能掉入

结论：经过多次测试得到了各分类垃圾桶对应的信号值分别为不可回收：814；可回收：854；有毒有害：934；其他：889。

（2）硬件软件联合调试

序号	1	2	3	4	5	6	6
第一层运行情况	正常	正常	正常	正常	正常	不正常	正常
第二层运行情况	不正常	不正常	不正常	不正常	不正常	正常	正常
Bug	垃圾没有掉落	垃圾没有掉落	垃圾掉落不对应桶	其他垃圾掉落不对应桶	有毒有害垃圾掉落不对应桶	第一层小舵机齿轮掉落	无

调整策略	增大 SD5 舵机摆角	继续增大 SD5 舵机摆角	调整小舵机摆角	调整小舵机摆角	调整小舵机角度	更换小舵机	无
调整后结果	不成功，依旧出现卡住的情况	成功，垃圾正常掉落	不成功，其他分类不对应桶	不成功，有毒有害垃圾不对应桶	成功	成功	无

结论：经过联合调试，得到了最优的程序以及硬件状态。
综上所述，本设计达到了设计要求。

九、 附录 1：参考文献

[1]郑泽宇、顾思宇.Tensorflow 实战 Google 深度学习框架[M].电子工业出版社，2017-2-10
[2]毛星.OpenCV3 编程入门[M].电子工业出版社，2015-2

十、 附录 2：作品操作说明

1. 使用步骤：打开开关，屏幕显示初始化中，待屏幕显示出一双眼睛，代表初始化结束，垃圾桶可以开始使用。正面对垃圾手持垃圾靠近垃圾桶盖，桶盖会自动打开，桶盖打开后可正常丢入垃圾，垃圾便能自动放入对应的分桶中。此过程中显示屏会显示垃圾的种类，分类，是否桶满等信息；
2. 日常清理：将垃圾桶第三层后盖打开后取出分桶，倒出垃圾后重新装入即可；
3. 紧急处理：在电源键旁边有一小圆孔，用小而尖的物体戳入即可复位。

十一、 附录 3：源代码

1. STM32 部分代码

```
int main(void)
 {
/***************初始化***************/
    noclear_flag=1;//不刷新
    delay_init();              //延时函数初始化
    NVIC_PriorityGroupConfig(NVIC_PriorityGroup_2);// 设置中断优先级分组 2
    TIM3_Int_Init(9999,7199);//10Khz 的计数频率，计数到 10000 为 1000ms  即 1s 的定时中断
    uart_init(115200);         //串口初始化为 115200
    LED_Init();                //初始化与 LED 连接的硬件接口
    LED0=1;
    LED1=1;
    LCD_Init();
    switch_Init();//光电开关初始化
    bling_Init();//继电器即闪光灯初始化
    EXTIX_Init();//按键中断初始化
    TIM2_PWM_Init(999,719); //pwm 初始化，频率 100hz：72M 先分频为 100k，再每 1000 次为一周期即 100hz
    TIM_SetCompare1(TIM2,929); //占空比低电平占比 93%  PA0  对应盖子舵机 初态关闭
//  TIM_SetCompare2(TIM2,849); //占空比低电平占比 85%  PA1
    TIM_SetCompare3(TIM2,839); //占空比低电平占比 84%  PA2  对应第一层阀门舵机  初态关闭
    TIM_SetCompare4(TIM2,849); //占空比低电平占比 95%  PA3  对应第二层分类舵机  初态竖直向下
/***************等待初始化***************/
    LCD_Show_HZ(0,80,0,0,2); //显示初始化
    LCD_Show_HZ(0+80,80,1,0,2);
    LCD_Show_HZ(80+80,80,2,0,2);
    LCD_Show_HZ(80+160,80,3,0,2);
    init_ok_flag=0;//初始化标志位为 0，等待树莓派初始化
    receive_flag=1;//允许接收数据
    while(!init_ok_flag); //初始化未完成时延时等待
    LCD_Clear(BLACK);
    noclear_flag=0;
/***************主程序***************/
  while(1)
    {
```

```
        if(noclear_flag==0)//不刷新标志为 0，即显示表情
        {
                seed_num*=1.7+1;
                srand(seed_num);      //设定随机数种子
                rand_num=rand()%10+1; //生成一个 1-10 的随机数
            //  printf("\r\n 随机数为:%d\r\n",rand_num);
                if((rand_num<=2 || smile_time!=0) && zy_time==0) //笑眼  延时 2
秒  条件：1、随机数符合或者在笑眼时间内  2、不在眨眼时间内
                {
                    if(smile_time==0)
                    {
                        smile_time=2;
                        LCD_Clear(BLACK); //首次清屏
                    }
                    LCD_Show_HZ(40,90,0,0,3);//左眼
                    LCD_Show_HZ(200,90,1,0,3);//右眼
                }
                else          //正常眼
                {
                    if(zy_time==0)
                        zy_time=3;
                    POINT_COLOR=LIGHTBLUE;
                    LCD_Draw_Circle(80,130,45);//眼睛
                    LCD_Draw_Circle(240,130,45);
            //  LCD_DrawLine(100,210,220,210);//嘴巴
                }
        }
        //光电开关
        if(GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_4)==0 && receive_flag==0)
//  if(GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_4)==0 )
        {
                delay_ms(20);
                if(GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_4)==0)
                {
                        TIM_SetCompare1(TIM2,749);//开盖
                        LED0=0;
                        PCout(2)=1;     //继电器即闪光灯打开
                        PCout(3)=1;
                        noclear_flag=1;//延时时不刷新
                        LCD_Clear(BLACK);
                        LCD_Show_HZ(40,90,0,0,3);//左眼
                        LCD_Show_HZ(200,90,1,0,3);//右眼
                        while(GPIO_ReadInputDataBit(GPIOA,      GPIO_Pin_4)==0);
```

```
                        delay_ms(1000);
                        delay_ms(1000);
                        delay_ms(1000);
                        TIM_SetCompare1(TIM2,939);//关盖
                        receive_flag=1;
                        printf("a");
                        noclear_flag=0;
                    }
                }
                else LED0=1;
            }
}
```

2. 深度学习模型核心代码

```
#!/usr/bin/env python
# -*- coding: utf-8 -*
import numpy as np
import tensorflow as tf
from tensorflow.python.platform import gfile
import glob,serial,time,os
import cv2
from os import system

BOTTLENECK_TENSOR_SIZE = 2048
BOTTLENECK_TENSOR_NAME = 'pool_3/_reshape:0'
JPEG_DATA_TENSOR_NAME = 'DecodeJpeg/contents:0'
import_BOTTLENECK_TENSOR_NAME = 'import/pool_3/_reshape:0'
import_JPEG_DATA_TENSOR_NAME = 'import/DecodeJpeg/contents:0'
CACHE_DIR = "/home/pi/Downloads/rubbish2.0/lable/"
INPUT_DATA = "/home/pi/Downloads/rubbish2.0/data/"          # 数据库存放地址

def create_image_lists(testing_percentage, validation_percentage):
    result = {}                                             # 保存所有图像。key 为类别名称。value 也是字典，存储了所有的图片名称
    sub_dirs = [x[0] for x in os.walk(INPUT_DATA)]          # 获取所有子目录
    is_root_dir = True
    for sub_dir in sub_dirs:
        if is_root_dir:                                     # 第一个目录为当前目录，需要忽略
            is_root_dir = False
            continue                                        # 繼續讀取子目錄
```

```
        # 获取当前目录下的所有有效图片
        extensions = ['jpg', 'jpeg', 'JPG', 'JPEG']

        file_list = []                                          # 存储所有图像
        dir_name = os.path.basename(sub_dir) # 获取路径的最后一个目录名字
        for extension in extensions:
            file_glob = os.path.join(INPUT_DATA, dir_name, '*.' + extension)
            file_list.extend(glob.glob(file_glob))
        if not file_list: continue

        # 将当前类别的图片随机分为训练数据集、测试数据集、验证数据集
        label_name = dir_name.lower()               # 通过目录名获取类别的名称

        training_images = []
        testing_images = []
        validation_images = []
        for file_name in file_list:
            base_name = os.path.basename(file_name)     # 获取该图片的名称

            # 随机划分数据
            chance = np.random.randint(100)     # 随机产生 100 个数代表百分
比
            if chance < validation_percentage:
                validation_images.append(base_name)
            elif chance < (testing_percentage + validation_percentage):
                testing_images.append(base_name)
            else:
                training_images.append(base_name)

        # 将当前类别的数据集放入结果字典
        result[label_name] = {
            'dir': dir_name,
            'training': training_images,
            'testing': testing_images,
            'validation': validation_images,
        }
    return result                                           # 返回整理好的所有数据

def uart():
    # ser.write("again\n".encode('utf-8'))   # 发送数据
    while True:
        count = ser.inWaiting()   # 获得接收缓冲区字符
```

```
        if count != 0:
            recv = ser.read(count)    # 读取内容,一次读 10 个字节
            bytes.decode(recv)    # 字节转成字符
            print(recv)    # 打印接收的数据
            return recv
        else:
            break
        ser.flushInput()    # 清空接收缓冲区
        time.sleep(0.1)    # 必要的软件延时

# 分类图片
def classify_photo(sess, jpeg_data_tensor, bottleneck_tensor):
    image_path = "/home/pi/Downloads/rubbish2.0/picture/opencv.jpeg"
    image_data = gfile.FastGFile(image_path, 'rb').read()
    #print (sess.run(jpeg_data_tensor,{jpeg_data_tensor:image_data}))
    #print (sess.run(bottleneck_tensor,{jpeg_data_tensor:image_data}))
    # """
    bottleneck_input                                                    =
sess.graph.get_tensor_by_name("BottleneckInputPlaceholder:0")
    final_tensor = sess.graph.get_tensor_by_name("final_training_ops/Softmax:0")
    bottleneck    =    sess.run(bottleneck_tensor,    feed_dict={jpeg_data_tensor:
image_data})
    class_result = sess.run(final_tensor, feed_dict={bottleneck_input: bottleneck})
    images_lists = create_image_lists(5, 5)
    a = np.squeeze(class_result)
    print(a)
    r = np.max(class_result)
    print(r)

    classes = ['bowl', 'paper', 'box', 'battery','milk-box', 'bottle']

    if r >= 0.80:
        kinds = int(np.argwhere(a==np.max(a)))
        print(kinds)
        print(classes[int(np.argwhere(a==np.max(a)))])
        if kinds == 3:
            ser.write("a12") #battery
        elif kinds == 2:
            ser.write("e12") #box
        elif kinds == 4:
            ser.write("c12") #milkbox
        elif kinds == 5:
            ser.write("d12") #bottle
```

```
            elif kinds == 0:
                ser.write("f12") #bowl
            elif kinds == 1:
                ser.write("b12") #paper
        else:
            print("此物品不在分类物品当中")
            ser.write("g12")

# 加载模型
saver =
tf.train.import_meta_graph("/home/pi/Downloads/rubbish2.0/module/modle/mode
l1.ckpt.meta")
with tf.Session() as sess:
        saver.restore(sess,
"/home/pi/Downloads/rubbish2.0/module/modle/model1.ckpt")
        bottleneck_tensor =
sess.graph.get_tensor_by_name(import_BOTTLENECK_TENSOR_NAME)
        jpeg_data_tensor =
sess.graph.get_tensor_by_name(import_JPEG_DATA_TENSOR_NAME)
        # all*
        ser = serial.Serial("/dev/ttyUSB0", 115200)    # 打开串口
        cap = cv2.VideoCapture(0)    #camera init
        ser.write("h12")
        while (1):
            key = uart()
            ret, frame = cap.read()    # 读取一帧
            cv2.namedWindow("capture", 0)
            cv2.resizeWindow("capture",640,480)
            cv2.imshow("capture", frame)
            cv2.waitKey(1)
            if key == 'a':
                cv2.imwrite("/home/pi/Downloads/rubbish2.0/picture/opencv.jpeg",
frame)    # 写入图片
                classify_photo(sess, jpeg_data_tensor, bottleneck_tensor)
            elif key == 'q':
                ser.write("shutdown-ing")
                ser.close()    # 释放串口
                sess.close()
                cap.release()    # 释放摄像头
                system("shutdown -t 0")
                break
            elif key== 'r':
                ser.write("reboot-ing")
```

```
ser.close()    # 释放串口
sess.close()
cap.release()    # 释放摄像头
system('reboot')
break
```