****

**华为杯**

**智能平衡小车设计报告**

**队伍名：\_\_\_\_\_\_\_\_\_Wind Driver\_\_\_\_\_\_\_\_\_\_\_\_**

**学院： 信息科学与工程学院**

**成员: 张成 李圭印 曹晓东 \_**

**2019年6月24日**

**目录**

**1.前言…………………………………………………………………………………**

**2.摘要…………………………………………………………………………………**

**3.方案设计……………………………………………………………………………**

**3.1总体方案设计………………………………………………………………**

**3.2方案选择与论证……………………………………………………………**

**4.硬件设计……………………………………………………………………………**

**4.1直流电机……………………………………………………………………**

**4.2驱动模块……………………………………………………………………**

**4.3循迹模块……………………………………………………………………**

**4.4控制模块……………………………………………………………………**

**4.5定位模块……………………………………………………………………**

**4.6避障模块……………………………………………………………………**

**4.7蓝牙模块……………………………………………………………………**

**4.8电量检测模块………………………………………………………………**

**4.9****摄像头跟踪模块……………………………………………………………**

**4.10语音识别模块 ……………………………………………………………**

**5.理论分析……………………………………………………………………………**

**5.1 传感器的接收情况…………………………………………………………**

**5.2 小车的状态分析……………………………………………………………**

**6.程序设计……………………………………………………………………………**

**6.1系统整体流程图……………………………………………………………**

**6.2各个部分的流程图…………………………………………………………**

**6.2.1 定时器的程序………………………………………………………**

**6.2.2传感器程序…………………………………………………………**

**7.结果分析与总结……………………………………………………………………**

**8.部分程序……………………………………………………………………………**

**前言**

**当今世界，随着计算机技术、控制技术、信息技术的快速发展，工业的生产和管理也都向着自动化、信息化、智能化方向发展。随着人们生活水平的提高，人们越来越希望全智能化的生活，智能化的东西可以按照预先设定的模式在一个环境里自动的运作，不需要人为的管理，为工业生产或日常生活提供很大的便利。在自动化生产线，智能仓库管理及物流配送等领域，当生产现场环境十分恶劣或者许多人工无法完成的搬运或者装卸时，机器人却能够适应这样恶劣的环境，这时候就需要智能循迹小车这样的机器来完成此类任务，基于现场和生活的实际需要，研究智能循迹小车的意义不言而喻。**

**自动循迹小车就是最简单的智能化产品，通过单片机的控制，能够让其沿着固定的轨道自动行驶，通过安卓APP可以控制小车的一切行为，而且可以获取传感器的所有数据，在APP上也可以通过摄像头看到小车的一切行为，并且实现了自动避障和跟踪的功能。通过对其的扩展，可以充分的应用在工厂自动化、军事领域、仓库管理、自动停车系统、智能玩具或民用服务等诸多领域，例如在自动仓库、码头、搬运、涂装等物流作业部门工作的物流小车就是在此基础上设计出来的。而且通过对这个课题的学习，通过理论与实践的结合，能够让自己对单片机的了解和应用进一步加深，另外，通过这次的设计，能够大大提高自己的动手能力，也更大的激发自己的兴趣。**

**国内外的研究概况: 国外智能车辆的始于上世纪50年代，它的发展历程大致可以分为以下三个阶段:第一阶段:1954年美国Barrett Electronic公司研究开发出了世界上第一台自主引导车系统，该系统只是一个运行在固定路线上的拖车式运货平台，但它却具有了智能车辆最基本的特征无人驾驶。第二阶段:从80年代中后期，在欧洲，普罗米修斯项目于1986年开始了在这个领域的探索，在美洲，美国于1995年成立了国家自动高速公路系统联盟，其目标之一就是研究发展智能车辆的可行性，并促进智能车辆技术进入实用化，在亚洲，日本在1996年成立了高速公路先进巡航辅助驾驶演剧协会，主要目的是研制自动车辆导航的方法，促进日本智能车辆的整体进步。进入80年代中期，设计和制造智能车辆的浪潮席卷了全世界，一大批世界著名的公司开始研制智能车辆平台。第三阶段:从90年代开始，智能车辆进入了深入、系统、大规模的研究阶段。最为突出的是，美国卡内基-梅陇大学机器人研究所完成了Navlab系列的自主车的研究，取得了显著的成就。**

**相比于国外，我国开展智能车辆技术方面的研究开始于20世纪80年代，而且大多数研究尚处于针对某个单项技术研究的阶段。虽然我国在智能车辆技术方面的研究总体上落后于发达国家，但是我国也取得了一系列的成果，主要有:中国第一汽车集团公司和国防科技大学于2003年研制成功了我国第一辆自主驾驶轿车;上海交通大学应用现代控制理论设计出了一种自动驾驶汽车模型，该模型在汽车系统的动力学建模的基础之上，设计了自动驾驶的专项系统，它能根据弯道的弯曲变化程度实时的计算出车辆的转向盘角度，控制车辆按照预设道路行驶;清华大学计算机系智能技术与系统国家重点实验室自1988年开始研制的THMR系列移动机器人取得了很大的成功。它兼有面向高速公路和一般道路的功能，目前已经能够在校园的非结构化道路环境下，进行道路跟踪和避障自主行驶。**

**摘 要**

**本设计是基于单片机控制的智能硬件小车系统，小车能够沿着宽约2CM的黑色轨迹行驶，利用安卓APP来控制小车的一切行为，获取安装在小车上的传感器的数据。本设计包括电源模块、蓝牙控制模块、循迹模块、电机驱动模块以及相关传感器模块。其中控制器模块以STM32单片机为控制核心，以亚克力板作为小车的车架，循迹模块是用红外光传感器来检测小车的运动轨迹，电机驱动模块使用的是L298N电机驱动芯片驱动4个带编码器的电机，利用霍尔传感器对电量进行检测，利用OpenMV H7模块实现跟踪功能，各个传感器将数据送给单片机，单片机再经过蓝牙模块给APP传送数据，从而实现APP对小车的实时监控。**

**本设计不仅给出了硬件设计流程，而且还给出了完整的硬件电路图和控制程序。在完成的既定的路线的基础上，我们也进行了很多的测试。**

**关键词:STM32 蓝牙模块 霍尔传感器OpenMV H7**

**一、设计方案**

**（一）、总体设计**

**设计制作一个智能小车，智能小车采用L298N电机驱动芯片驱动4个带编码器的电机，安卓APP控制小车的一切行为，获取传感器数据，可以听声音，看录像（获取摄像头的数据），自动避障和跟踪。**

**(二)、方案选择与比较**

**1、电动机的选择**

**方案一:**

**采用直流电机。直流电机转动力矩大，响应快速，体积小，重量轻，直流电动机具有优良的调速特性，调速平滑、方便，调整范围广;过载能力强，能承受频繁的冲击负载，可实现频繁的无级快速启动、制动和反转,能满足各种不同的特殊运行要求，价格便宜。**

**方案二:**

**采用步进电机。步进电机是一种将电脉冲信号转换成角位移或线位移的精密执行原件。控制方便，体积小，灵活性和可靠性高，具有瞬时启动和急速停止的优越性，比较适合本系统控制精度高的特点。但步进电机的抖动比较大，输出力矩较低，随转速的升高而下降，且在较高转速时会急剧下降，其转速较低，不适用于小车等有一定速度要求的系统，价格还比较昂贵，所以这里不采用此方案。**

**由于直流电机价格便宜、控制简单，因此本设计用方案一。**

**2、电动机驱动模块的选择**

**方案一:**

**采用电阻网络或数字电位器调整电动机的分压，从而达到调速目的。但是电阻网络只能实现有级调速，而数字电阻的元器件价格比较昂贵，且可能存在干扰。更主要的问 在于一般电动机的电阻比较小，但电流很大，分压不仅会降低效率，而且实现很困难。**

**方案二:**

**采用继电器对电动机的开与关进行控制，通过控制开关的切换速度实现对小车的速度进行调整。这个电路的优点是电路较为简单，缺点是继电器的响应时间长，易损坏，寿命较短，可靠性不高。**

**方案三:**

**采用专用电机驱动芯片L298N作为电机驱动芯片。** **L298N模块是2路的H桥驱动，所以可以同时驱动两个电机，接法如图所示使能ENA ENB之后，可以分别从IN1 IN2输入PWM信号驱动电机1的转速和方向，可以分别从IN3 IN4输入PWM信号驱动电机2的转速和方向。**

**基于以上的分析，建议电动机驱动电路选择方案三。**

**3、电源模块**

**方案一:**

**电脑USB串口供电。能直接为单片机提供稳定的+5V直流电压。USB串口线又容易得到。但需要很长的线，这样导致无法在室外工作。**

**方案二:**

**用7.2V充电电池组作为小车供电电源。经7805稳压后给单片机供电，而7.2V电压可直接接在驱动芯片上作为两个直流电机的驱动电压。在不超过单片机工作电压范围的情况下，又能驱动直流电机。这个电源结构简单，价格便宜，容易得到，而且能够重复使用。**

**方案三:**

**采用4节普通5号电池作为小车的供电电源。刚买的5号电池测得电压为1.7V，4节就是6.8V，单片机需要5V电源，因此用7805稳压到5V后供电，但是其放电电流不大，导致电动机转速很慢，而且在使用过程中，其电压会明显降低，普通5号电池会降到1.4V以下，这样导致经过7805稳压后电压小于5V，完全无法带动整个系统正常工作，因此放弃该方案。**

**综上所述，选择方案三作为小车电源模块，经济实惠。**

**4、控制器的选择**

**方案一:**

**STM32单片机作为系统的控制器。在STM32F105和STM32F107互连型系列微控制器之前，[意法半导体](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=502986&ss_c=ssc.citiao.link" \t "_blank)已经推出STM32基本型系列、增强型系列、USB基本型系列、互补型系列；新系列产品沿用增强型系列的72MHz处理频率。内存包括64KB到256KB闪存和 20KB到64KB嵌入式SRAM。新系列采用LQFP64、LQFP100和LFBGA100三种封装，不同的封装保持引脚排列一致性，结合STM32平台的设计理念，开发人员通过选择产品可重新优化功能、存储器、性能和引脚数量，以最小的硬件变化来满足个性化的应用需求。方案二:**

**采用FPGA作为系统的主控制器。FPGA可以实现各种复杂的逻辑功能，规模大，集成度高，体积小，稳定性好，IO口资源丰富，易于进行功能扩展，处理速度快，常用于大规模实时性要求较高的系统，但价格高，编程实现难度大。**

**综上，我们采用方案一。**

**二、硬件设计**

**1、直流减速电机**

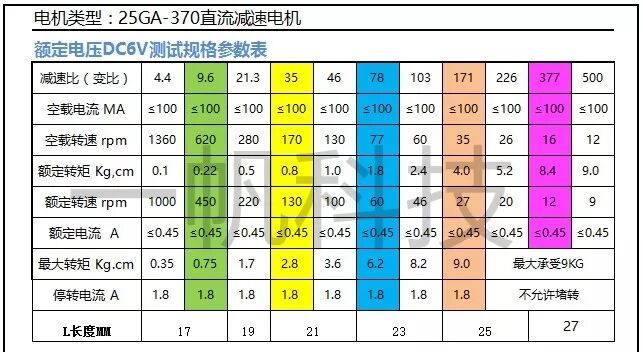
**(1)、基本原理**

**直流减速电机可以根据两端的电压的大小或者利用pwm波和正负实现速度和方向的改变。当两个电机以相同速度运行时小车前进或者后退;当两个小车以不同速度运行时小车转向，例如当左轮比右轮转的快时，小车向右转向。当左轮比右轮转的慢的时候小车向左转。当小车以相同速度反方向转时，小车可以实现原地转向。**

**(2)、实物图如图所示:**

****

**相关参数:**



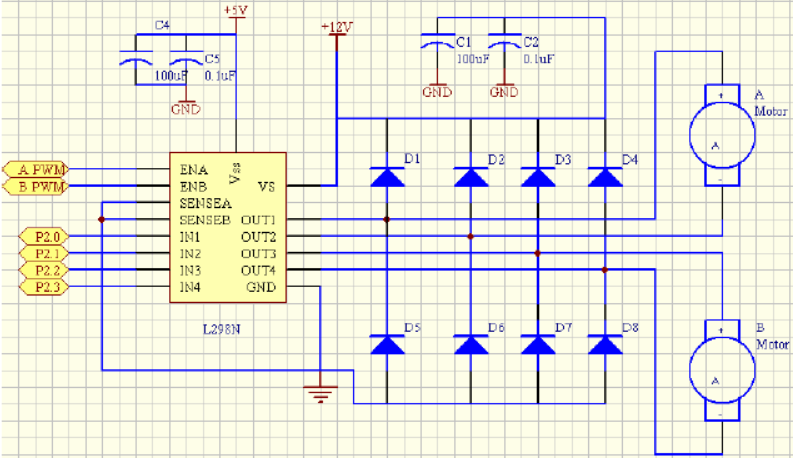
**(3)、性能介绍**

**直流碳刷价格便宜驱动简单，但是受限于其电机转轴以及外侧轮胎的加工精度，在驱动时，不容易实现完全相同的速度。因此对于小车而言，在安装好后需要首先进行软件校准是一件十分重要的事情。让两个电机直线行驶状态时转速尽量相同，有利于减少循迹时鸭子步和摆头的情况。**

**2、驱动模块**

**(1)、驱动电路**

**由于电机的电流较大，单片机不能直接驱动电机运行，因此需要配合电机驱动模块使用。我组采用的是以L298N芯片为核心的驱动电路。**

****

**(2)、相关参数介绍**

**1. 驱动芯片：L298N双H桥直流电机驱动芯片**

**2. 非门芯片：IA、IB的高低电平分别控制A 、B两个电机正反转**

**3. 驱动部分端子供电范围VMS：＋5V～＋46V**

**4. 驱动部分峰值电流Io：2A**

**5. 逻辑部分端子供电范围VCC：＋5V**

**6. 逻辑部分端子供电范围VIN：＋6.5V～＋12V（稳压之后给芯片VCC供电）**

**7. 逻辑部分工作电流范围：0～20mA**

**8. 控制信号输入电压范围：**

**低电平：－0.3V≤Vin≤1.5V**

**高电平：2.3V≤Vin≤Vss**

**9. 使能信号输入电压范围：**

**低电平：－0.3≤Vin≤1.5V（控制信号无效）**

**高电平：2.3V≤Vin≤Vss（控制信号有效）**

**10. 最大功耗：25W（温度T＝75℃时）**

**11. 正常工作温度：－25℃～＋130℃**

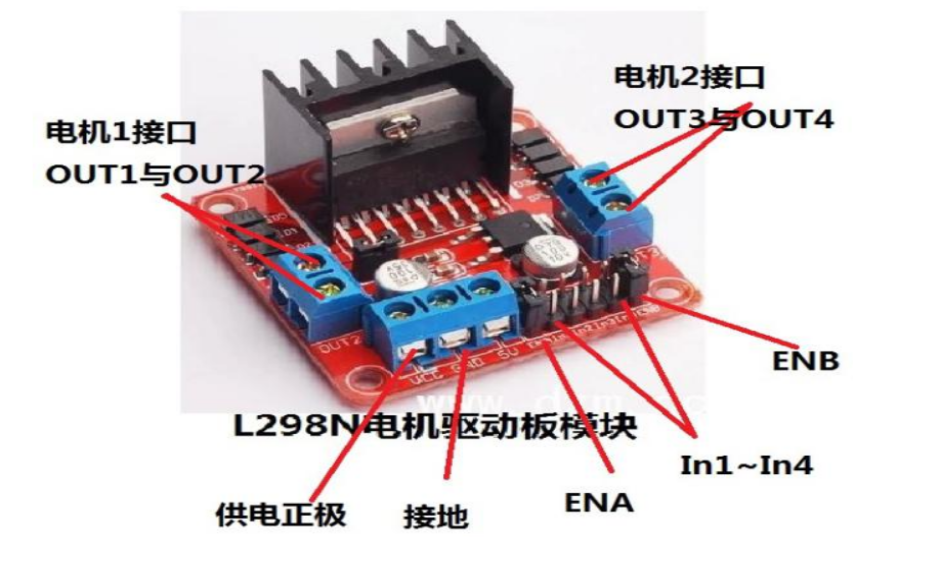
**12. 驱动板尺寸：58mm×54mm**

**13. 驱动板重量：35g**

图5

**(3)、L298N的基本原理**

**L298N是专用驱动集成电路，属于H桥集成电路，与L293D的差别是其输出电流增大，功率增强。其输出电流为2A，最高电流4A，最高工作电压50V，可以驱动感性负载，如大功率直流电机，步进电机，电磁阀等，特别是其输入端可以与单片机直接相联，从而很方便地受单片机控制。当驱动直流电机时，可以直接控制步进电机，并可以实现电机正转与反转，实现此功能只需改变输入端的逻辑电平。**

****

**3、循迹模块**

**（1） 模块描述**

**4路循迹传感器模块**

**当模块检测到前方障碍物信号时，电路板上红色指示灯点亮，同时 OUT 端口持续输出低电平信号,该模块检测距离 2～60cm，检测角度 35°，检测距离可 以通过电位器进行调节，顺时针调电位器，检测距离增加；逆时针调电位器，检 测距离减少。**

**2、传感器属于红外线反射探测,因此目标的反射率和形状是探测距离的关键。其 中黑色探测距离最小,白色最大。**

**3、传感器模块输出端口 OUT 可直接与单片机 IO 口连接即可，也可以直接驱动一 个 5V 继电器模块或者蜂鸣器模块；连接方式：VCC-VCC;GND-GND;OUT-IO**

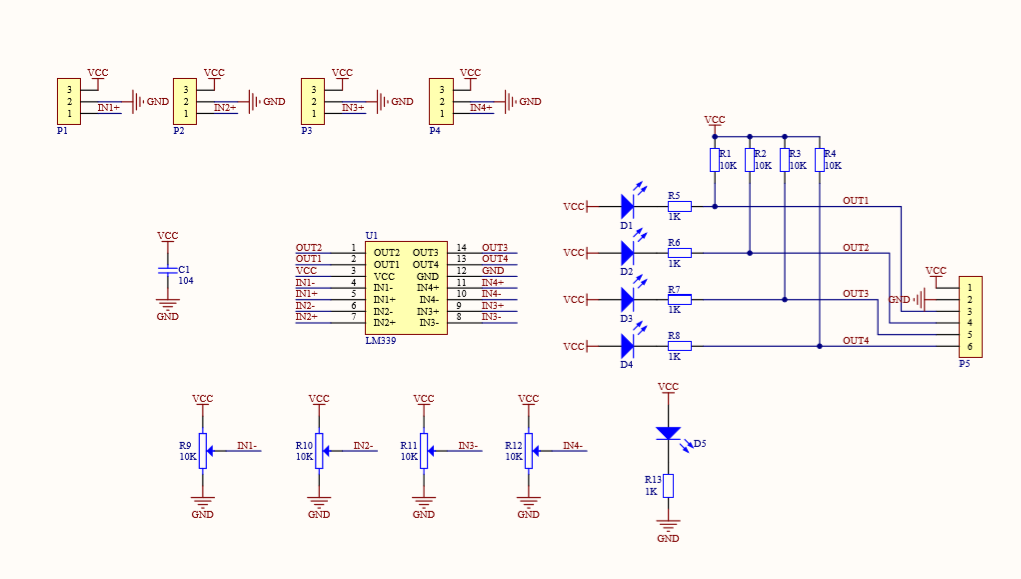
**4、比较器采用 LM339，工作稳定；**

**5、可采用 3.3V-5V 直流电源对模块进行供电。当电源接通时，绿色电源指示灯 点亮；**

**6、具有 3mm 的螺丝孔，便于固定、安装；**

**7、尺寸大小：中控板 42mm×38mm×12mm（长×宽×高） 小板 25mm×12mm×12mm（长×宽×高）**

**（2）主控板图**

****

**4、控制模块**

**(1)、基本原理**

**一个单片机应用系统的硬件电路设计包含有两部分内容:一是系统扩展，即单片机内部的功能单元，如ROM)RAM)I/O 口)定时/记数器)中断系统等能量不能满足应用系统的要求时，必须在片外进行扩展，选择适当的芯片，设计相应的电路。二是系统配置，既按照系统功能要求配置外围设备，如键盘显示器)打印机)A/D)D/A转换器等，要设计合适的接口电路。STM32 单片机是把那些作为控制应用所必需的基本内容都集成在一个尺寸有限的集成电路芯片上。**

**内核：32 位 高性能 ARM Cortex-M3 处理器。时钟：高达 72M,实际还**

**可以超频一点。单周期乘法和硬件除法。**

**（2）IO 口：STM32F103ZET6: 144 引脚 112 个 IO，大部分 IO 口都耐 5V(模**

**拟通道除外)，支持调试：SWD 和 JTAG，SWD 只要 2 根数据线**

**（3）存储器容量：512K FLASH，64K SRAM**

**（4）时钟，复位和电源管理：**

**①2.0~3.6V 电源和 IO 电压**

**②上电复位，掉电复位和可编程的电压监控**

**③强大的时钟系统**

**-4~16M 的外部高速晶振**

**-内部 8MHz 的高速 RC 振荡器**

**-内部 40KHz 低速 RC 振荡器，看门狗时钟**

**-内部锁相环（PLL，倍频），一般系统时钟都是外部或者内部高速**

**时钟经过 PLL 倍频后得到**

**- 外部低速 32.768K 的晶振，主要做 RTC 时钟源**

**（5）低功耗：**

**-睡眠，停止和待机三种低功耗模式**

**-可用电池为 RTC 和备份寄存器供电**

**（6）AD:**

**-3 个 12 位 AD（多达 21 个外部测量通道）**

**-转换范围：0-3.6V（参考电源电压）**

**-内部通道可以用于内部温度测量**

**-内置参考电压**

**（7）DA:2 个 12 位 DA**

**（8）DMA:12 个 DMA 通道（7 通道 DMA1，5 通道 DMA2），支持外设：定时器，**

**ADC,DAC，SDIO,I2S,SPI,I2C,和 USART**

**（9）定时器：多达 11 个定时器**

**-4 个通用定时器**

**-2 个基本定时器**

**-2 个高级定时器**

**-1 个系统定时器**

**-2 个看门狗定时器**

**（10）通信接口：多达 13 个通信接口**

**-2 个 I2C 接口**

**-5 个串口**

**-3 个 SPI 接口**

**-1 个 CAN2.0**

**-1 个 USB FS**

**5、定位模块**

**重力加速度陀螺仪传感器MPU-6050**

**MPU6050模块是InvenSense公司推出的一款低成本的6轴传感器模块，包括三轴加速度，三轴角速度。其体积小巧，用途非常广。做平衡小车，四轴飞行器，飞行鼠标等等，都是必不可少而且是最优的传感器解决方案。**

**引脚图：**

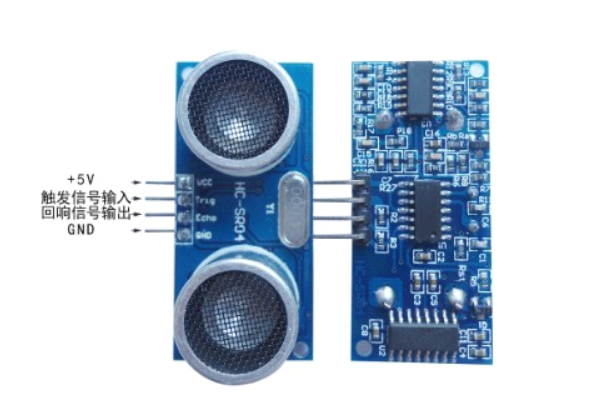
****

**6、避障模块**

**方案一：用超声波传感器进行避障。超声波传感器的原理是：超声波由压电陶瓷超声波传感器发出后，遇到障碍物便反射回来，再被超声波传感器接收。超声波传感器在避障的设计中被广泛应用。但是超声波传感器需要40KHz的方波信号来工作，因为超声波传感器对工作频率要求较高，偏差在1％内，所以用模拟电路来做方波发生器比较难以实现。因此我们考虑其它的方案。**

**方案二：用漫反射式光电开关进行避障。光电开关的工作原理是根据光线发射头发出的光束，被物体反射，其接收电路据此做出判断反应，物体对红外光由同步回路选通而检测物体的有无。当有光线反射回来时，输出低电平。当没有光线反射回来时，输出高电平。考虑到本系统只需要检测障碍物，没有十分复杂的环境。为了使用方便，便于操作和调试，我们最终选择了方案一**

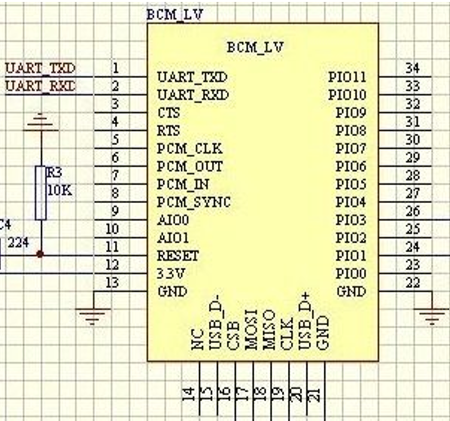
**实物图如下图所示**

****

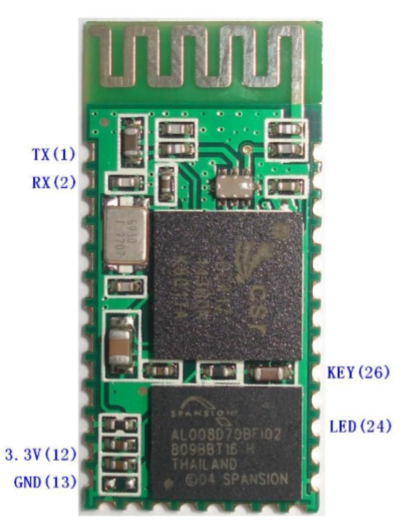
**7、蓝牙模块**

**HC-05是主从一体化的蓝牙串口模块。（供电电压 3.3V~3.6V）**

**原理图**

****

**实物图**

****

**8、电量检测模块**

**电池剩余电量监测模块SOC(SEU)**

**该电控单元可用于电动车的电池组剩余电量监测。控制单元采用英飞凌的xc2xxx系列单片机，充放电电流检测采用英飞凌的TLE4997霍尔传感器。实现方式是通过在存储器中存储特定电池的放电特性数据和校正参数，并采集电池组的开路电压、充放电电流运用剩余电量计算算法计算电池组的剩余电量，同时监测电池组的温度和循环充放电次数根据特定电池的特性对剩余电量进行校正。**

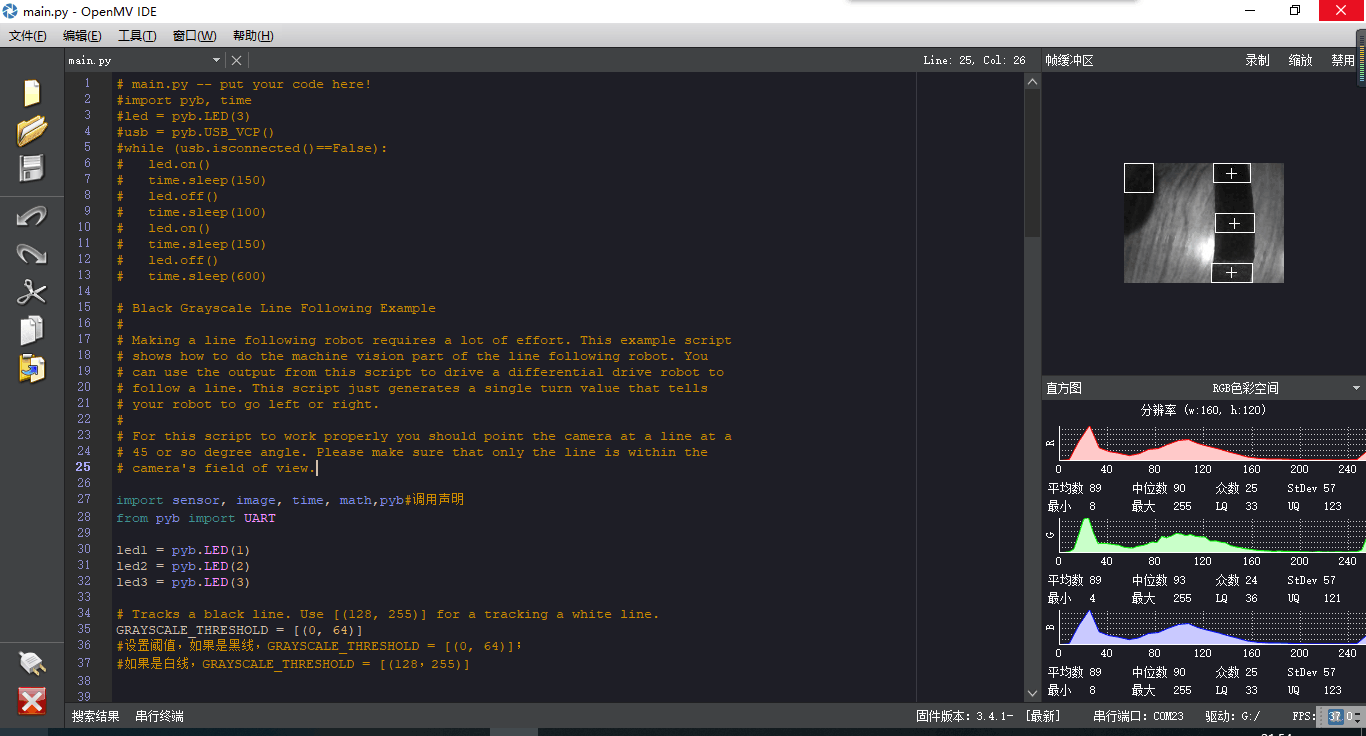
**9、摄像头跟踪模块**

**OpenMV项目旨在通过开发开源的低成本机器[视觉](https://www.cirmall.com/circuits/tags/1806)[摄像机](https://www.cirmall.com/circuits/tags/5041)，为业余爱好者和制造商提供机器视觉。第一代OpenMV摄像机基于STM的[STM32](https://www.cirmall.com/circuits/tags/372)F ARM Cortex-M MCU和Omnivision OV7725传感器。OpenMV摄像机可以在Python3中进行编程，并附有大量的图像处理功能，如面部检测和跟踪，关键点描述符，彩色斑点跟踪，QR和条形码支持，AprilTags，GIF和MJPEG记录等等。**

**OpenMV摄像机板内置RGB和红外LED，用于编程和视频流的USB FS，uSD插座和I / O头，可以分解PWM，UART，SPI和I2C。此外，OpenMV还支持使用诸如WiFi，BLE，Thermal（FIR）和LCD屏蔽等I / O头的扩展模块（屏蔽）。**

****

**OpenMV配备了专门用于支持OpenMV摄像机的跨平台IDE（基于QT创建者）。IDE允许查看帧缓冲区，访问传感器控制，上传脚本并通过串行通过USB（或WiFi / BLE（如果可用））在相机上运行它们。人脸识别程序如图**

****

**10、语音识别模块**

**语音识别模块主要由语音识别芯片和相关附属电路组成，其主要功能是识别语音指令，并根据识别出的指令信息通过串口发送相应的指令编码。**

**目前常用的语音识别芯片有不可修改的 OTP 芯片、可以重复烧录 flash 语音芯片和支持 MP3 码的语音芯片。考虑到成本与应用环境的需求，我们选择支持MP3 解码的 LD3320 语音识别芯片，这是一颗基于非特定人的语音识别（SI-ASR）芯片，它集成了高精度的 A/D 和 D/A 接口，可以实现语音识别、声控和人机对话功能，并且识别的关键词语列表可以动态编辑。考虑到智能小车工作时要不断地运动，所以用麦克风作为输入接口显然不是最佳选择，因此，本设计采用咪头作为输入接口。**

**三、理论分析**

**循迹传感器工作状态**

**当黑色轨迹线在传感器中间时，传感器2接受到黑线，即**

**(注:1表示接收到信号，0表示没有接收到返回的信号)**

**传感器1 传感器2 传感器3**

**0 1 0**

**小车直行**

**转弯情况：**

**传感器1 传感器2 传感器3**

**1 0 0**

**小车左转**

**传感器1 传感器2 传感器3**

**0 0 1**

**小车右转**

**传感器1 传感器2 传感器3**

**1 1 0**

**小车左小转**

**传感器1 传感器2 传感器3**

**0 1 1**

**小车右小转**

**特殊情况**

**传感器1 传感器2 传感器3**

**1 1 1**

**小车停止**

**四、程序设计**

**1、系统整体流程图**

开始

单片机初始化

读取传感器状态

传感器是否检测到数据

**否**

数据是否符合要求

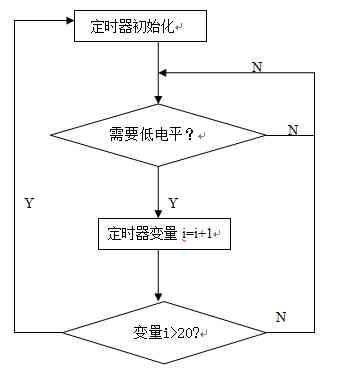
修正数据

**是**

上传数据

**2、部分模块程序**

**（1）、定时器程序**

****

**（2）循迹程序**

**小车开始循迹**

**检测两侧传感器信号**

**判断当前状态**

**是**

**是否触发两侧传感器**

**按传感器情况进行转向控制**

**否**

**说明进入了直线状态**

**记数延时保证跑完**

**直行**

**五、实验结果分析**

**1、小车功能**

**智能小车能按照规定路线进行循迹，通过各种传感器模块对声音，图像进行了采集，并且通过蓝牙将传感器检测的数据上传给手机，用户通过手机上的app 能对小车进行实时的监控与控制。**

**2、传感器功能**

**循迹光电传感器能正确地使小车按照规定的路线行驶，超声波避障模块能使小车及时绕开障碍物。蓝牙模块与手机进行实时通信，并在手机上实时显示小车剩余的电量。 通过语音识别模块，用户可以对小车进行语音控制，小车也通过openMV模块，对检测到的物体进行图像处理，及时上传给手机app上。**

1. **部分程序**

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**“华为杯”智能循迹小车**

**河北科技大学“逐风者”队**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

**行走部分**

**//大于零 前进 小于零 后退**

**void set\_motor\_forward\_back\_speed(int lspeed,int rspeed)**

**{**

**if(lspeed>=0)**

**{**

**TIM3->CCR1 =lspeed;**

**TIM3->CCR2 =0;**

**TIM4->CCR1 =lspeed;**

**TIM4->CCR2 =0;**

**}**

**else**

**{**

**TIM3->CCR1 = 0;**

**TIM3->CCR2 = -lspeed;**

**TIM4->CCR1 = 0;**

**TIM4->CCR2 = -lspeed;**

**}**

**if(rspeed>=0)**

**{**

**TIM3->CCR3 =rspeed;**

**TIM3->CCR4 =0;**

**TIM4->CCR3 =rspeed;**

**TIM4->CCR4 =0;**

**}**

**else**

**{**

**TIM3->CCR3 = 0;**

**TIM3->CCR4 = -rspeed;**

**TIM4->CCR3 = 0;**

**TIM4->CCR4 = -rspeed;**

**}**

**}**

**//右转: 均大于0**

**//左转: 均小于0**

**void set\_motor\_left\_right\_speed(int lspeed,int rspeed)**

**{**

**if(lspeed>=0)**

**{**

**TIM3->CCR1 =lspeed;**

**TIM3->CCR2 =0;**

**TIM4->CCR1 =0;**

**TIM4->CCR2 =lspeed;**

**}**

**else**

**{**

**TIM3->CCR1 = 0;**

**TIM3->CCR2 = -lspeed;**

**TIM4->CCR1 = -lspeed;**

**TIM4->CCR2 = 0;**

**}**

**if(rspeed>=0)**

**{**

**TIM3->CCR3 =0;**

**TIM3->CCR4 =rspeed;**

**TIM4->CCR3 =rspeed;**

**TIM4->CCR4 =0;**

**}**

**else**

**{**

**TIM3->CCR3 = -rspeed;**

**TIM3->CCR4 = 0 ;**

**TIM4->CCR3 = 0;**

**TIM4->CCR4 = -rspeed ;**

**}**

**}**

**//循迹控制程序**

**void TRAC\_Control(void)**

**{**

**if(LEFT\_LITTLE\_TURN) //左小转**

**{**

**yaw\_switch=0; //标记yaw发生变化 需要重新定义初值**

**set\_motor\_left\_right\_speed(-LITTLE\_TURN\_SPEED,-LITTLE\_TURN\_SPEED);**

**}**

**else if(LEFT\_BIGGER\_TURN) //左大转**

**{**

**yaw\_switch=0;**

**set\_motor\_left\_right\_speed(-BIGGER\_TURN\_SPEED,-BIGGER\_TURN\_SPEED);**

**}**

**else if(RIGHT\_LITTLE\_TURN) //右小转**

**{**

**yaw\_switch=0;**

**set\_motor\_left\_right\_speed(LITTLE\_TURN\_SPEED,LITTLE\_TURN\_SPEED);**

**}**

**else if(RIGHT\_BIGGER\_TURN) //右大转**

**{**

**yaw\_switch=0;**

**set\_motor\_left\_right\_speed(BIGGER\_TURN\_SPEED,BIGGER\_TURN\_SPEED);**

**}**

**else if(STOP)**

**{**

**yaw\_switch=0;**

**set\_motor\_left\_right\_speed(0,0);**

**set\_motor\_forward\_back\_speed(0,0);**

**}**

**else if(FORWARD) //前进**

**{**

**if(yaw\_switch==0)**

**{**

**yaw\_offset=yaw; //记录转向后的初始yaw**

**yaw\_switch=1;**

**}**

**if(pid\_control\_mode!=FORWARD\_BACK\_PID\_CONTROL)**

**{**

**set\_motor\_forward\_back\_speed(FORWARD\_SPEED,FORWARD\_SPEED);**

**}**

**else**

**{**

**pid\_control\_mode=FORWARD\_BACK\_PID\_CONTROL;**

**}**

**}**

**else //其余情况下 慢速前进**

**{**

**yaw\_switch=0;**

**if(\_\_fabs(pitch)<5)**

**{**

**set\_motor\_forward\_back\_speed(SLOW\_SPEED,SLOW\_SPEED);**

**}**

**}**

**}**

**PID控制部分**

**void TIM2\_IRQHandler(void) //TIM2中断**

**{**

**if (TIM\_GetITStatus(TIM2, TIM\_IT\_Update) != RESET) //检查指定的TIM中断发生与否:TIM 中断源**

**{**

**TIM\_ClearITPendingBit(TIM2, TIM\_IT\_Update ); //清除TIMx的中断待处理位:TIM 中断源**

**js\_tim2++;**

**if(Init\_Flag)**

**{**

**time++;**

**if(\_\_fabs(pitch)>10&&walk\_status==NORMAL)**

**{**

**walk\_status=SPECIAL; //开始上跷跷板**

**pid\_control\_mode|=FORWARD\_BACK\_PID\_CONTROL;**

**}**

**if(walk\_status==SPECIAL)**

**{**

**if(pitch\_count<10000)**

**pitch\_count=0;**

**else if(**

**pitch\_count++;**

**if(pitch\_count>10000)**

**pid\_control\_mode&=!FORWARD\_BACK\_PID\_CONTROL;**

**}**

**}**

**if(pid\_control\_mode&FORWARD\_BACK\_PID\_CONTROL)**

**{**

**PID\_Control(pitch,pitch\_offset,&CrossSeesaw\_Pitch\_Position);**

**PID\_Control(y\_gyro,CrossSeesaw\_Pitch\_Position.pid\_out,&CrossSeesaw\_Pitch\_Speed);**

**VAL\_LIMIT(CrossSeesaw\_Pitch\_Speed.pid\_out,-MAX\_SPEED,MAX\_SPEED); //pid输出限幅**

**set\_motor\_forward\_back\_speed(CrossSeesaw\_Pitch\_Speed.pid\_out,CrossSeesaw\_Pitch\_Speed.pid\_out);**

**}**

**if(pid\_control\_mode&LEFT\_RIGHT\_PID\_CONTROL)**

**{**

**if(FORWARD) //只在前进时对左右进行PID控制**

**{**

**PID\_Control(yaw,yaw\_offset,&CrossSeesaw\_Yaw\_Position);**

**PID\_Control(z\_gyro,CrossSeesaw\_Yaw\_Position.pid\_out,&CrossSeesaw\_Yaw\_Speed);**

**VAL\_LIMIT(CrossSeesaw\_Yaw\_Speed.pid\_out,-MAX\_SPEED,MAX\_SPEED); //pid输出限幅**

**set\_motor\_left\_right\_speed(CrossSeesaw\_Yaw\_Speed.pid\_out,CrossSeesaw\_Yaw\_Speed.pid\_out);**

**}**

**}**

**}**

**else**

**{**

**if(js\_tim2%300==0)**

**led=!led;**

**}**

**}**

**}**

**图像识别部分**

**# Black Grayscale Line Following Example**

**#**

**# Making a line following robot requires a lot of effort. This example script**

**# shows how to do the machine vision part of the line following robot. You**

**# can use the output from this script to drive a differential drive robot to**

**# follow a line. This script just generates a single turn value that tells**

**# your robot to go left or right.**

**#**

**# For this script to work properly you should point the camera at a line at a**

**# 45 or so degree angle. Please make sure that only the line is within the**

**# camera's field of view.**

**import sensor, image, time, math#调用声明**

**from pyb import UART**

**# Tracks a black line. Use [(128, 255)] for a tracking a white line.**

**GRAYSCALE\_THRESHOLD = [(0, 64)]**

**#设置阈值，如果是黑线，GRAYSCALE\_THRESHOLD = [(0, 64)]；**

**#如果是白线，GRAYSCALE\_THRESHOLD = [(128，255)]**

**# Each roi is (x, y, w, h). The line detection algorithm will try to find the**

**# centroid of the largest blob in each roi. The x position of the centroids**

**# will then be averaged with different weights where the most weight is assigned**

**# to the roi near the bottom of the image and less to the next roi and so on.**

**ROIS = [ # [ROI, weight]**

**(0, 100, 160, 20, 0.7), # You'll need to tweak the weights for you app**

**(0, 050, 160, 20, 0.3), # depending on how your robot is setup.**

**(0, 000, 160, 20, 0.1)**

**]**

**#roi代表三个取样区域，（x,y,w,h,weight）,代表左上顶点（x,y）宽高分别为w和h的矩形，**

**#weight为当前矩形的权值。注意本例程采用的QQVGA图像大小为160x120，roi即把图像横分成三个矩形。**

**#三个矩形的阈值要根据实际情况进行调整，离机器人视野最近的矩形权值要最大，**

**#如上图的最下方的矩形，即(0, 100, 160, 20, 0.7)**

**# Compute the weight divisor (we're computing this so you don't have to make weights add to 1).**

**weight\_sum = 0 #权值和初始化**

**for r in ROIS: weight\_sum += r[4] # r[4] is the roi weight.**

**#计算权值和。遍历上面的三个矩形，r[4]即每个矩形的权值。**

**# Camera setup...**

**sensor.reset() # Initialize the camera sensor.**

**sensor.set\_pixformat(sensor.GRAYSCALE) # use grayscale.**

**sensor.set\_framesize(sensor.QQVGA) # use QQVGA for speed.**

**sensor.skip\_frames(30) # Let new settings take affect.**

**sensor.set\_auto\_gain(False) # must be turned off for color tracking**

**sensor.set\_auto\_whitebal(False) # must be turned off for color tracking**

**#关闭白平衡**

**clock = time.clock() # Tracks FPS.**

**while(True):**

**clock.tick() # Track elapsed milliseconds between snapshots().**

**img = sensor.snapshot() # Take a picture and return the image.**

**uart = UART(3,19200)**

**uart.init(19200,bits=8,parity=None,stop=1)#init with given parameters**

**centroid\_sum = 0**

**#利用颜色识别分别寻找三个矩形区域内的线段**

**for r in ROIS:**

**blobs = img.find\_blobs(GRAYSCALE\_THRESHOLD, roi=r[0:4], merge=True)**

**# r[0:4] is roi tuple.**

**#找到视野中的线,merge=true,将找到的图像区域合并成一个**

**#目标区域找到直线**

**if blobs:**

**# Find the index of the blob with the most pixels.**

**most\_pixels = 0**

**largest\_blob = 0**

**for i in range(len(blobs)):**

**#目标区域找到的颜色块（线段块）可能不止一个，找到最大的一个，作为本区域内的目标直线**

**if blobs[i].pixels() > most\_pixels:**

**most\_pixels = blobs[i].pixels()**

**#merged\_blobs[i][4]是这个颜色块的像素总数，如果此颜色块像素总数大于 #most\_pixels，则把本区域作为像素总数最大的颜色块。更新most\_pixels和largest\_blob**

**largest\_blob = i**

**# Draw a rect around the blob.**

**img.draw\_rectangle(blobs[largest\_blob].rect())**

**img.draw\_rectangle((0,0,30, 30))**

**#将此区域的像素数最大的颜色块画矩形和十字形标记出来**

**img.draw\_cross(blobs[largest\_blob].cx(),**

**blobs[largest\_blob].cy())**

**centroid\_sum += blobs[largest\_blob].cx() \* r[4] # r[4] is the roi weight.**

**#计算centroid\_sum，centroid\_sum等于每个区域的最大颜色块的中心点的x坐标值乘本区域的权值**

**center\_pos = (centroid\_sum / weight\_sum) # Determine center of line.**

**#中间公式**

**# Convert the center\_pos to a deflection angle. We're using a non-linear**

**# operation so that the response gets stronger the farther off the line we**

**# are. Non-linear operations are good to use on the output of algorithms**

**# like this to cause a response "trigger".**

**deflection\_angle = 0**

**#机器人应该转的角度**

**# The 80 is from half the X res, the 60 is from half the Y res. The**

**# equation below is just computing the angle of a triangle where the**

**# opposite side of the triangle is the deviation of the center position**

**# from the center and the adjacent side is half the Y res. This limits**

**# the angle output to around -45 to 45. (It's not quite -45 and 45).**

**deflection\_angle = -math.atan((center\_pos-80)/60)**

**#角度计算.80 60 分别为图像宽和高的一半，图像大小为QQVGA 160x120.**

**#注意计算得到的是弧度值**

**# Convert angle in radians to degrees.**

**deflection\_angle = math.degrees(deflection\_angle)**

**#将计算结果的弧度值转化为角度值**

**A=deflection\_angle**

**print("Turn Angle: %d" % int (A))#输出时强制转换类型为int**

**#print("Turn Angle: %d" % char (A))**

**# Now you have an angle telling you how much to turn the robot by which**

**# incorporates the part of the line nearest to the robot and parts of**

**# the line farther away from the robot for a better prediction.**

**print("Turn Angle: %f" % deflection\_angle)**

**#将结果打印在terminal中**

**uart\_buf = bytearray([int (A)])**

**#uart\_buf = bytearray([char (A)])**

**#uart.write(uart\_buf)#区别于uart.writechar是输出字符型，这个函数可以输出int型**

**uart.write(uart\_buf)**

**uart.writechar(0x41)#通信协议帧尾**

**uart.writechar(0x42)**

**time.sleep(10)#延时**

**print(clock.fps()) # Note: Your OpenMV Cam runs about half as fast while**

**# connected to your computer. The FPS should increase once disconnected.**

**安卓APP主要程序**

**public class MainActivity extends AppCompatActivity {  
 //save our header or result  
 private AccountHeader headerResult = null;  
 private Drawer result = null;  
 private MiniDrawer miniResult = null;  
 private CrossfadeDrawerLayout crossfadeDrawerLayout = null;  
  
 //bluetooth  
 public static final int MESSAGE\_STATE\_CHANGE = 1;  
 public static final int MESSAGE\_READ = 2;  
 public static final int MESSAGE\_WRITE = 3;  
 public static final int MESSAGE\_DEVICE\_NAME = 4;  
 public static final int MESSAGE\_TOAST = 5;  
  
 public static final String DEVICE\_NAME = "device\_name";  
 public static final String TOAST = "toast";  
  
 public static final byte[] out = {'t'};  
 public static char direction = 'w';  
  
 private static final int REQUEST\_CONNECT\_DEVICE = 1;  
 private static final int REQUEST\_ENABLE\_BT = 2;  
  
 // private TextView mTitle;  
 private ListView mConversationView;  
 private EditText mOutEditText;  
 private Button mSendButton;  
  
 private Button up\_button;  
 private Button left\_button;  
 private Button down\_button;  
 private Button right\_button;  
 private Button stop\_button;  
 private Button fire\_button;  
 private Button random\_button;  
  
 private String mConnectedDeviceName = null;  
 private ArrayAdapter<String> mConversationArrayAdapter;  
 private StringBuffer mOutStringBuffer;  
 private BluetoothAdapter mBluetoothAdapter = null;  
 private BluetoothChatService mChatService = null;  
  
  
 //gravity  
 // 感应器管理器  
 private SensorManager sensorMgr;  
 // 得到加速感应器  
 Sensor sensor;  
 // 定义各坐标轴上的重力加速度  
 private float x, y, z;  
 boolean graisopen = false;  
  
  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_sample\_dark\_toolbar);  
  
 //Remove line to test RTL support  
 //getWindow().getDecorView().setLayoutDirection(View.LAYOUT\_DIRECTION\_RTL);  
  
 // Handle Toolbar  
 Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);  
 setSupportActionBar(toolbar);  
  
 //Bluetooth  
 mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();  
  
 if (mBluetoothAdapter == null)  
 {  
 Toast.makeText(this, "ǰֻ֧.",  
 Toast.LENGTH\_LONG).show();  
 finish();  
 return;  
 }  
  
 //gravity  
 // 得到当前手机传感器管理对象  
 sensorMgr = (SensorManager) getSystemService(SENSOR\_SERVICE);  
  
 // 加速重力感应对象  
 sensor = sensorMgr.getDefaultSensor(Sensor.TYPE\_ACCELEROMETER);  
  
 // 实例化一个监听器  
 SensorEventListener lsn = new SensorEventListener() {  
 // 实现接口的方法  
  
 public void onSensorChanged(SensorEvent e) {  
 // 得到各轴上的重力加速度  
 x = e.values[SensorManager.DATA\_X];  
 y = e.values[SensorManager.DATA\_Y];  
 z = e.values[SensorManager.DATA\_Z];  
 if(graisopen&&mChatService!= null){  
 // 在标题处显示出来  
 setTitle("重力感应模式 X:" + (int)x + "," + "Y:" + (int)y + ","+ "Z:" + (int)z);  
 //Toast.makeText(MainActivity.this, "X轴上的重力加速度为:" + x + "," + "Y轴上的重力加速度为:" + y + "," + "Z轴上的重力加速度为:" + z, Toast.LENGTH\_SHORT).show();  
  
 if(z>5){  
 //前  
 out[0] = 'w';  
 direction = 'w';  
 sendCommand(out);  
 }else if(z<-5){  
 //后  
 out[0] = 's';  
 direction = 's';  
 sendCommand(out);  
 }else {  
 //停  
 sendCommand(out);  
 out[0] = 't';  
 sendCommand(out);  
 }  
  
 if(x>6){  
 //左  
 out[0] = 'a';  
 sendCommand(out);  
 }else if(x<-6){  
 //右  
 out[0] = 'd';  
 sendCommand(out);  
 }  
 }  
  
 }  
  
 public void onAccuracyChanged(Sensor s, int accuracy) {  
 }  
 };  
 // 注册listener，第三个参数是检测的精确度  
 sensorMgr.registerListener(lsn, sensor, SensorManager.SENSOR\_DELAY\_GAME);  
  
  
 FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);  
 fab.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
// Snackbar.make(view, "Replace with your own action", Snackbar.LENGTH\_LONG)  
// .setAction("Action", null).show();  
 remote\_ctrl();  
 graisopen = !graisopen;  
 setTitle("遥控触碰模式");  
 }  
 });  
  
  
 // Create a few sample profile  
 // NOTE you have to define the loader logic too. See the CustomApplication for more details  
 final IProfile profile = new ProfileDrawerItem().withName("小车遥控器").withEmail("https://github.com/hongfeiyucode").withIcon(R.mipmap.ic\_launcher);  
 final IProfile profile2 = new ProfileDrawerItem().withName("红绯鱼").withEmail("https://github.com/hongfeiyucode").withIcon(R.drawable.conan);  
  
 // Create the AccountHeader  
 headerResult = new AccountHeaderBuilder()  
 .withActivity(this)  
 .withTranslucentStatusBar(true)  
 .withHeaderBackground(R.drawable.header)  
 .addProfiles(  
 profile, profile2  
 )  
 .withSavedInstance(savedInstanceState)  
 .build();  
  
  
 //Create the drawer  
 result = new DrawerBuilder()  
 .withActivity(this)  
 .withToolbar(toolbar)  
 .withDrawerLayout(R.layout.crossfade\_material\_drawer)  
 .withHasStableIds(true)  
 .withDrawerWidthDp(72)  
 .withGenerateMiniDrawer(true)  
 .withAccountHeader(headerResult) //set the AccountHeader we created earlier for the header  
 .addDrawerItems(  
 new PrimaryDrawerItem().withName(R.string.drawer\_item\_first).withDescription("右下按钮开启重力感应").withIcon(MaterialDesignIconic.Icon.gmi\_car).withIdentifier(1),  
 new PrimaryDrawerItem().withName(R.string.drawer\_item\_second).withIcon(MaterialDesignIconic.Icon.gmi\_directions\_run).withIdentifier(2),  
 new PrimaryDrawerItem().withName(R.string.drawer\_item\_third).withIcon(MaterialDesignIconic.Icon.gmi\_truck).withIdentifier(3),  
 new PrimaryDrawerItem().withName(R.string.drawer\_item\_fourth).withIcon(MaterialDesignIconic.Icon.gmi\_airplane).withIdentifier(4),  
 new PrimaryDrawerItem().withName(R.string.drawer\_item\_fifth).withIcon(MaterialDesignIconic.Icon.gmi\_search).withIdentifier(5),  
 new PrimaryDrawerItem().withName(R.string.drawer\_item\_sixth).withIcon(FontAwesome.Icon.faw\_eye).withIdentifier(6),  
 new DividerDrawerItem(),  
 new SecondaryDrawerItem().withName(R.string.drawer\_item\_seventh).withIcon(FontAwesome.Icon.faw\_user\_secret).withIdentifier(7).withSelectable(false)  
 ) // add the items we want to use with our Drawer  
 .withOnDrawerItemClickListener(new Drawer.OnDrawerItemClickListener() {  
 @Override  
 public boolean onItemClick(View view, int position, IDrawerItem drawerItem) {  
  
 up\_button = (Button) findViewById(R.id.button\_up);  
 up\_button.setEnabled(false);  
 down\_button = (Button) findViewById(R.id.button\_down);  
 down\_button.setEnabled(false);  
 right\_button = (Button) findViewById(R.id.button\_right);  
 right\_button.setEnabled(false);  
 left\_button = (Button) findViewById(R.id.button\_left);  
 left\_button.setEnabled(false);  
 stop\_button = (Button) findViewById(R.id.button\_stop);  
 stop\_button.setEnabled(false);  
 fire\_button = (Button) findViewById(R.id.button\_fire);  
 fire\_button.setEnabled(false);  
  
 if (drawerItem.getIdentifier() == 1) {  
 Toast.makeText(MainActivity.this, ((Nameable) drawerItem).getName().getText(MainActivity.this), Toast.LENGTH\_SHORT).show();  
 remote\_ctrl();  
 return true;  
 }else  
 if (drawerItem.getIdentifier() == 2) {  
 Toast.makeText(MainActivity.this, ((Nameable) drawerItem).getName().getText(MainActivity.this), Toast.LENGTH\_SHORT).show();  
 auto\_ctrl\_1m();  
 return true;  
 }else  
 if (drawerItem.getIdentifier() == 3) {  
 Toast.makeText(MainActivity.this, ((Nameable) drawerItem).getName().getText(MainActivity.this), Toast.LENGTH\_SHORT).show();  
 auto\_ctrl\_3m();  
 return true;  
 }else  
 if (drawerItem.getIdentifier() == 4) {  
 Toast.makeText(MainActivity.this, ((Nameable) drawerItem).getName().getText(MainActivity.this), Toast.LENGTH\_SHORT).show();  
 auto\_avoid\_obstacle();  
 return true;  
 }else  
 if (drawerItem.getIdentifier() == 5) {  
 Toast.makeText(MainActivity.this, ((Nameable) drawerItem).getName().getText(MainActivity.this), Toast.LENGTH\_SHORT).show();  
 Intent serverIntent = new Intent(MainActivity.this, DeviceListActivity.class);  
 startActivityForResult(serverIntent, REQUEST\_CONNECT\_DEVICE);  
 return true;  
 }else  
 if (drawerItem.getIdentifier() == 6) {  
 Toast.makeText(MainActivity.this, ((Nameable) drawerItem).getName().getText(MainActivity.this), Toast.LENGTH\_SHORT).show();  
 ensureDiscoverable();  
 return true;  
 }else  
 if (drawerItem.getIdentifier() == 7) {  
// new LibsBuilder()  
// .withFields(R.string.class.getFields())  
// .withActivityStyle(Libs.ActivityStyle.DARK)  
// .start(MainActivity.this);  
 } else {  
 if (drawerItem instanceof Nameable) {  
 Toast.makeText(MainActivity.this, ((Nameable) drawerItem).getName().getText(MainActivity.this), Toast.LENGTH\_SHORT).show();  
 }  
 }  
  
 return false;  
 }  
 })  
 .withSavedInstance(savedInstanceState)  
 .withShowDrawerOnFirstLaunch(true)  
 .build();  
  
 //get out our drawerLyout  
 crossfadeDrawerLayout = (CrossfadeDrawerLayout) result.getDrawerLayout();  
  
 //define maxDrawerWidth  
 crossfadeDrawerLayout.setMaxWidthPx(DrawerUIUtils.getOptimalDrawerWidth(this));  
 //add second view (which is the miniDrawer)  
 MiniDrawer miniResult = result.getMiniDrawer();  
 //build the view for the MiniDrawer  
 View view = miniResult.build(this);  
 //set the background of the MiniDrawer as this would be transparent**