目录

1 任务分析

1.1 题目简述

在本次综合设计中，我选择的题目是“智能平衡小车设计”。本题目旨在应用电子元件和功能模块制作一款两轮自平衡小车，实现运动控制、通信、避障和循迹等功能。

该设计的基本要求包括：

- 小车采用单片机或嵌入式系统作为主控模块
- 小车采用两轮式驱动，能够自主保持平衡
- 小车可以通过蓝牙模块与手机通信，实现 APP 控制小车运行
- 小车运动速度可通过 APP 或按钮调整

该设计的拓展要求包括：

- 小车自主运行，遇到障碍物自动回避
- 通过 WIFI 模块控制小车运行
- 小车通过超声波模块检测前方障碍物距离并显示距离数值
- 小车通过红外对管实现在白底黑线背景下的自动巡线功能

1.2 成果要求

根据设计题目要求，本次设计确定的验收目标包括：

- 两轮平衡小车实体
- 小车电路设计原理图
- 小车 PCB 设计图
- 小车主控源代码
- 手机 APP 源代码
- 设计报告

1.3 分工安排

本次综合设计由个人独立完成，工作内容包括：设计方案，选择元件和模块，学习各模块的使用，整合模块基本调用代码，绘制原理图，绘制 PCB，打板和焊接，开发主控程序，开发 APP，调参，测试以及撰写报告等。

2 平衡控制原理

平衡控制是本次综合设计的关键功能，接下来将从平衡控制的运动分析和算法两大角度介绍。

2.1 运动分析

以现实生活中杂技演员保持木棒在自己指尖上直立为例子，当木棒倾斜或出现倾斜的趋势时，演员需要通过手掌移动抵消木棒的倾斜角度和趋势，从而保持木棒的直立，并且，木棒倾斜的角度或倾斜的趋势越大，手的运动幅度也相应增大。

保持物体平衡的过程可以看作是通过运动对抗运动的过程，事实上就是控制理论中的负反馈机制。

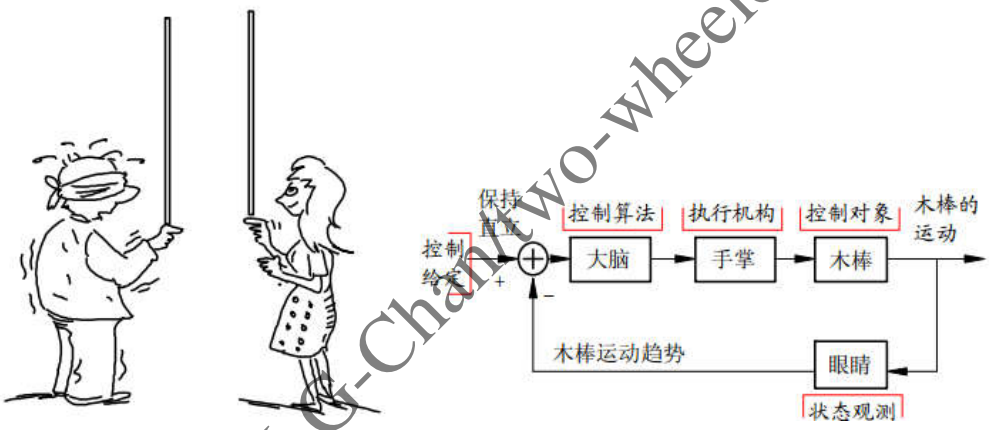

图 1 保持木棒直立的负反馈控制例子

小车保持平衡也是通过负反馈来实现的。我们可以将小车的主体视为只会在轮子滚动的方向上发生倾斜的“木棒”，将小车的车轮看成是手掌，当小车向左倾斜时，车轮应该加速向左运行以“接住”小车主体；当小车向右倾斜时，车轮应该加速向右运行以“接住”小车主体。

图 2 通过车轮运动保持小车主体平衡

将小车视为倒立摆，通过数学建模和受力分析，我们可以得出让小车尽快垂直稳定下来需要通过控制电机加速产生恢复力和阻尼力，其中回复力：

$$F = mg\sin\theta - ma\cos\theta \approx mg\theta - mk_1\theta$$

加上阻尼力后：

$$F = mg\theta - mk_1\theta - mk_2\theta'$$

定义控制车轮产生的加速度a和小车倾角θ、角加速度θ'，小车直立时满足关系：

$$a = k_1\theta + k_2\theta'$$

其中比例参数$k_1 > g, k_2 > 0$时，直立小车可稳定。

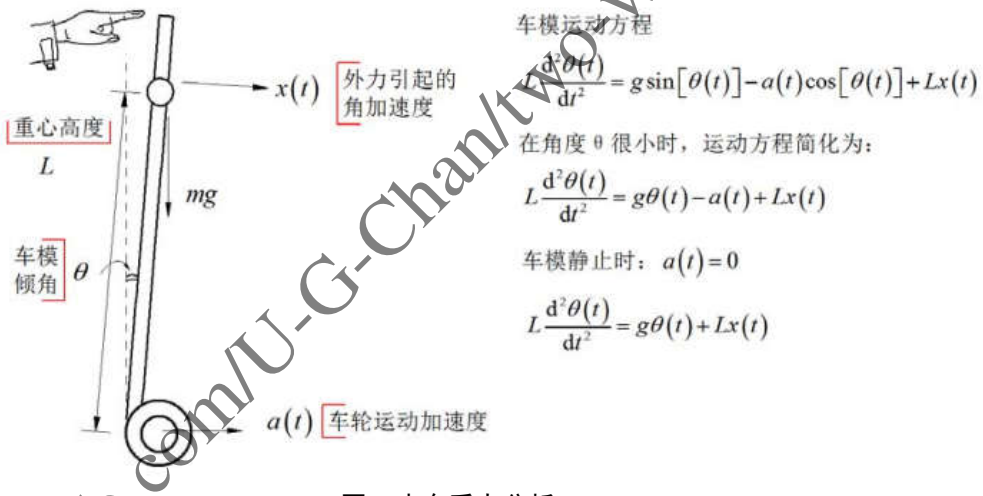

图 3 小车受力分析

总而言之，控制车模直立稳定的条件如下：

- 能够精确测量车模倾角 θ 的大小和角速度 θ '的大小；
- 可以控制车轮的加速度。

为此，我们需要可以测得小车运动姿态以及电机转速的传感器模块才能实现对小车平衡的反馈调节。

2.2 PID 算法

假设我们可以获取小车每一时刻的倾角、角速度和速度等运动变量，我们还需要确定反馈系数，并通过运算得出正确反馈量才能完成平衡控制。为此我们需要使用 PID 算法。

PID 是经典的自动控制算法，在工业控制系统中广泛应用。P、I、D 的意义分别是比例（Proportional）、积分（Integral）、微分（Derivative），其算法的核心是对误差的数学运算，通过计算设定点（期望值）和实际输出之间的误差，产生一个控制量，用于调整被控制系统的输入，以减小误差并将系统的输出稳定在设定点上。

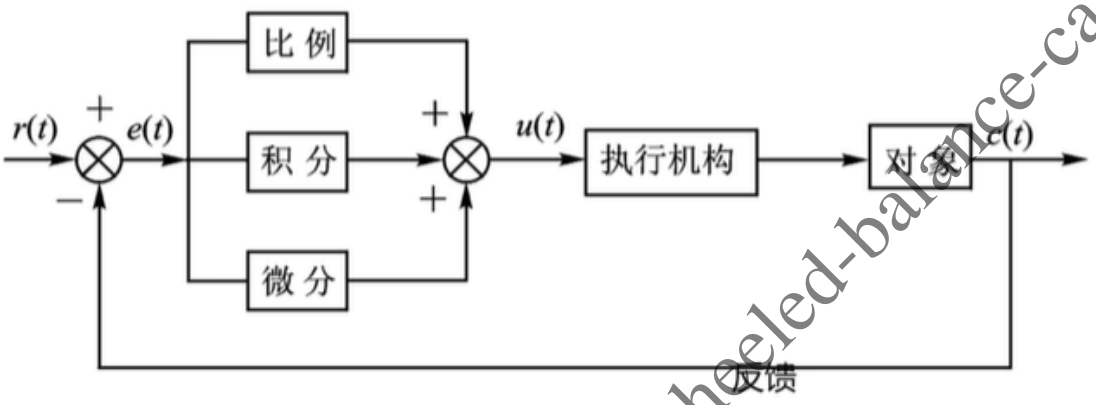

图 4 PID 算法流程图

连续 PID 公式如下：

$$u(t) = K_p\left[e(t) + \frac{1}{T_1}\int_0^t e(t)dt + T_D\frac{de(t)}{dt}\right]$$

由于程序更擅长处理离散计算，我们可以由以上公式简化为

位置式 PID：

$$u_k = K_p \cdot e_k + K_i \cdot \sum_{j=0}^{k} e_j + K_d \cdot (e_k - e_{k-1})$$

增量式 PID：

$$\Delta u_k = K_p \cdot (e_k - e_{k-1}) + K_i \cdot e_k + K_d \cdot (e_k - 2 \cdot e_{k-1} + e_{k-2})$$

位置式 PID 更关注绝对量的变化，增量式 PID 更关注趋势的变化，两者的选择取决于我们需要控制的结果，比如我们希望控制小车的倾角则使用位置式 PID，希望控制小车的速度则用增量式 PID。

观察 PID 控制算法的公式，我们可以划分为比例项$K_p \cdot e_k$、积分项$K_i \cdot \sum_{j=0}^{k} e_j$和微分项$K_d \cdot (e_k - e_{k-1})$。各项的作用特点如下：

(1) 比例项

- K_p越大，系统响应越快，越快达到目标值。
- K_p过大会导致系统产生超调和振荡
- 仅通过比例项无法消除静态误差

图 5 比例调节

(2) 积分项

- K_i越大，系统响应越快达到目标值，引入积分项可消除误差
- K_i过大会导致系统产生较大的超调和振荡
- 随着时间的增加，积分项变大，系统响应会变慢

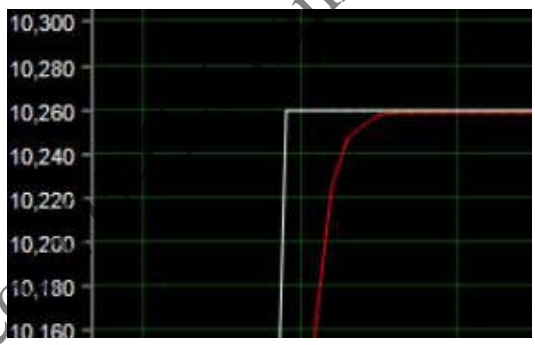

图 6 积分调节

(3) 微分项

- K_d或变化趋势越大，微分环节作用越强，对超调和振荡的抑制越强
- K_d过大会引起系统的不稳定，容易引入高频噪声

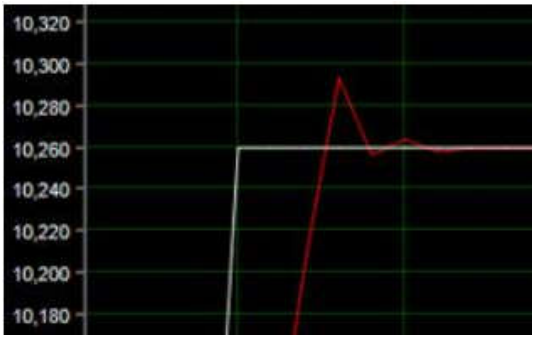

图 7 微分控制

PID 控制过程不一定三项都会参与，比如在我们的项目中，小车平衡只用 PD 控制，轮子速度的控制只采用 PI 控制。

2.3 直立控制

为了保持小车直立，小车的车身往哪边倾斜，我们就需要控制轮子往哪边运动。

小车直立需要一个快速的响应，并且在平衡车模型中小车受到地球重力影响几乎不可能产生稳态误差，因为小车只要没有达到机械中值就会倾倒，所以这里直接省略掉了积分控制。

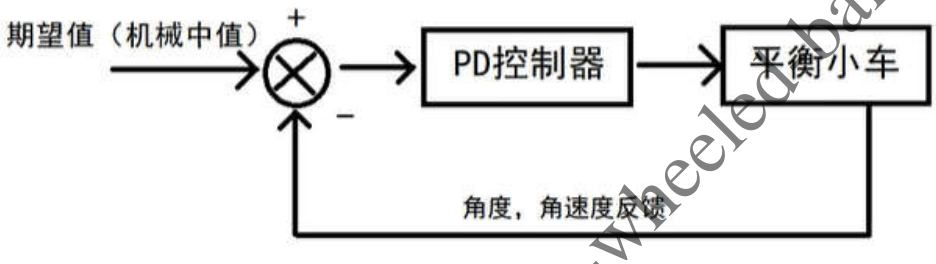

图 8 小车直立控制回路

其中，机械中值是因为受到小车组装等因素影响，小车的真实平衡位置可能不是 0 度，我们需要自行测试小车平衡时的倾角。此角度才是我们期望的角度。图为只有小车在机械中值的时候才能保持平衡，实现直立。将倾角和角速度量代入 PID 公式，我们可以得到：

$$e_k = ZERO - \theta$$

$$e_k - e_{k-1} = \theta' - \theta$$

由于离散化测量，$\theta' - \theta$即是小车倾倒的角速度ω，所以直立环的反馈量：

$$PWM = K_p * (ZERO - \theta) + K_d * \omega$$

2.4 速度控制

我们希望小车在保持平衡的同时，还能以一定速度运动。在直立控制过程中，小车的车身往哪边倾斜，我们就需要控制轮子往哪边运动，小车倾角越大，保持平衡所需的速度越大。所以，速度控制的就是在直立控制中的角度控制。

将速度控制中的期望值（机械中值）替换为速度控制量的输出，直立控制回路根据速度控制量来调整倾斜角度， 倾角的改变导致小车速度变化，

速度化量再反馈到速度控制器，其原理可参考下图：

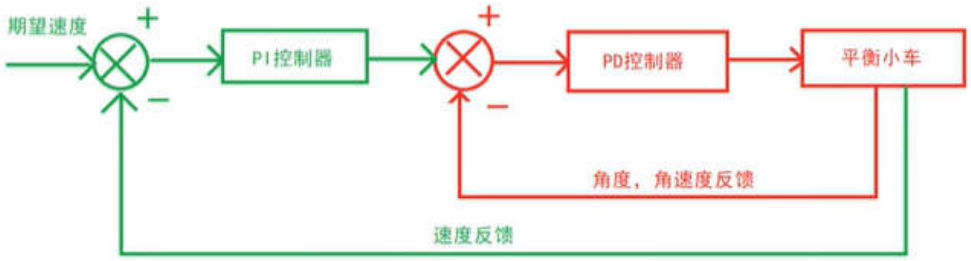

图 9 速度控制回路

由于速度调节应该是平缓连续的，所以我们应该舍去微分控制，采用 PI 控制器。速度控制回路采用了串级 PID 控制原理，外环称为速度环，内环称为直立环。

根据 PI 控制回路可以得出速度控制的公式：

$$output = K_p \cdot (encoder\_set - encoder) + K_i \cdot \sum_{j=0}^{k} e_j$$

其中$encoder_set - encoder$在离散测量中表示车轮转速，$output$是 PI 控制器的输出。

将$(output + ZERO)$式代入直立环中的 ZERO，可得速度控制公式：

$$PWM = K_p * K_p'(encoder_{set} - encoder) + K_p * K_i \cdot \sum_{j=0}^{k} e_j + K_p * ZERO - K_p * \theta + K_d * \omega$$

其中$K_p * K_p'(encoder_{set} - encoder) + K_p * K_i \cdot \sum_{j=0}^{k} e_j$正好是速度环，$K_p * ZERO - K_p * \theta + K_d * \omega$正好是直立环。因此这个双环 PID 的控制代码相当于把两个环的输出结果求和后传入电机驱动器。

2.5 转向控制

小车两个车轮的转速不同即可实现转向。在直立环和速度环的基础上叠加转向环的 PWM 即可实现。在转向控制用有两种控制需求，一种是抑制转向，另一种是期望转向。

由于各自外界因素的影响，即使向小车的两个电机即使输入的 PWM 是

完全相同的也会产生速度差，导致小车即使没有转向控制也会在运动过程中发生偏转，所以我们希望通过转向控制抑制这种偏差，实现小车直线行走。对于期望转向情况，我们可以直接对两电机 PWM 进行加减操作。

转向控制不要求特别精确，因此我们在期望转向时只是用 P 调节，抑制转向时只是用 D 调节。D 调节我们直接使用陀螺仪的 Z 辅角速度。

抑制转向公式：

$$output = K_d * Gyro_Z$$

期望转向公式：

$$output = K_p * Angle_Z$$

在上述公式中，微分项可以抑制控制量的抖动，削弱因为左右轮的速度偏差导致的偏转，比例项可以根据传输的期望转角提供两轮速度差，引起小车转动，当$Angle_Z$为 0 时即可保持小车直行。

3 模块选择与介绍

3.1 主控模块

综合考虑实现智能平衡小车功能的复杂程度，个人应用熟练度等因素，本设计采用了 STM32F103 系列芯片作为小车的主控芯片。

STM32F103 系列芯片因其成本效益高、性能稳定、资源丰富以及易于上手等特点受到许多工程师和爱好者的青睐,具备性价比高、开发生态系统丰富、外设丰富、开发环境丰富和兼容性高等特点，对于本项目的设计来说极具性价比。

在板形选择方面，本项目采用了 STM32F103C8T6 最小系统板。一方面，由于本次设计并不计划批量生产，所以直接在 PCB 上绘制最小系统并不能节省多少成本，反而因为需要购买散装外围元件和设计失误等原因造成额外成本；另一方面，STM32F103C8T6 最小系统板可以直接插在面包板上，通过跳线连接其他模块，方便前期验证和调试。

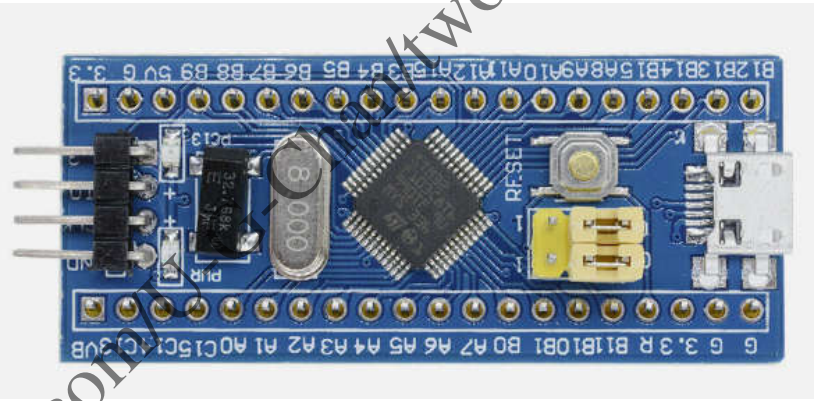

图 10 STM32F103C8T6 最小系统板

3.2 直流电机模块

本项目采用一对 MG310 电机作为两轮平衡小车的运动执行机构。该款电机具有功率体积比高，稳定性高等特点。

这款电机的参数如下：

表 1 MG310 电机参数

属性	参数
编码器类型	霍尔(磁感应)
减速比	1:20
额定电压	7.4V

额定扭矩	0.4kgf.cm
额定电流	≤500mA
额定转速	400 士 13%RPM
堵转扭矩	1.5 士 kgf.cm
堵转电流	<2mA

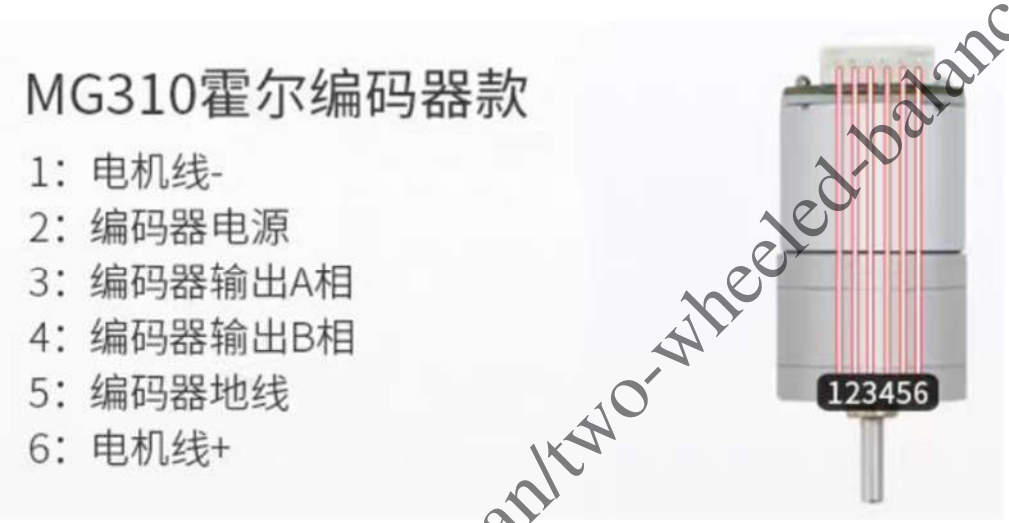

图 11 MG310 电机接线说明

这款电机自带霍尔编码器，可以通过磁电转换将输出轴上的机械几何位移量转换成脉冲或数字量。编码器有 AB 相输出，所以不仅可以测速，还可以辨别转向。我们只需给编码器电源 5V 供电，在电机转动的时候即可通过 AB 相输出方波信号。

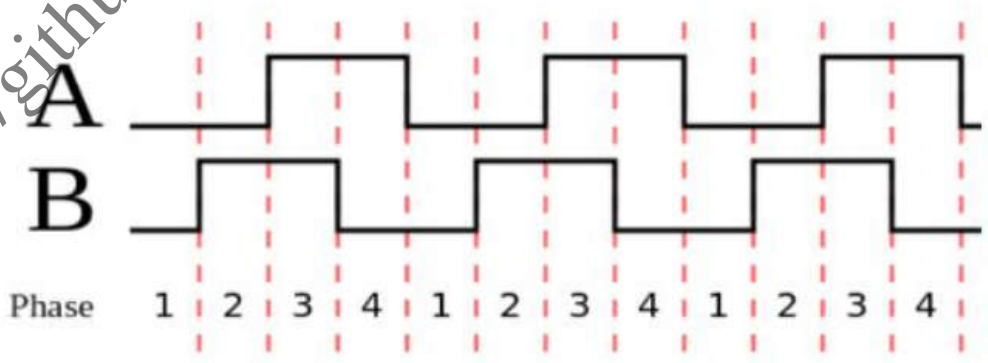

图 12 编码器输出波形图

编码器自带了上拉电阻，所以无需外部上拉，可以直接 STM32F103 自带编码器接口，使用硬件计数，进而得出车轮的位移和速度。

3.3 电机驱动模块

由于直流电机是大电流感性负载，而单片机 I0 的带负载能力较弱，所以我们需要功率放大器件。

本项目采用经典直流电机驱动模块 TB6612FNG 作为直流电机驱动器件，该模块具有大电流 MOSFET-H 桥结构,双通道电路输出,可同时驱动 2 个电机。它无需外加散热片，外围电路简单，只需外接电源滤波电容就可以直接驱动电机，利于减小系统尺寸。对于 PWM 信号输入频率范围，高达 100 kHz 的频率更是足以满足平衡车的控制的需求。

TB6612FNG 模块的接线如下图所示：

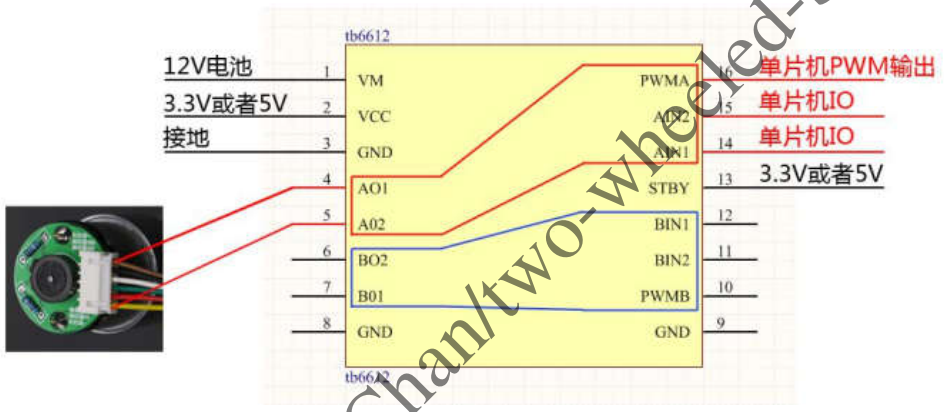

图 13 TB6612FNG 模块的接线示例

其中 STBY 接高电平时使能驱动模块，AIN1、AIN2 控制电机的正转反转，PWMA 端接收单片机的 PWM 输出以实现电机转速的控制，AO1、AO2 将直流控制量输出到电机的+和-端。

3.4 陀螺仪模块

在 2.1 节中提到，要想控制小车平衡，就必去要获取小车的姿态。而小车当前的姿态实际上可以通过三轴加速度和三轴陀螺仪的原始测量数据计算得出，为此本项目选择 YH-MPU6050 模块进行测量。

MPU6050 芯片内部的 DMP 模块 (Digital Motion Processor 数字运动处理器)，可对传感器数据进行滤波、融合处理，直接通过 IIC 接口向主控器输出姿态解算后的数据，降低主控器的运算量。其姿态解算频率最高可达 200Hz，非常适合用于对姿态控制实时要求较高的领域。

图 14 MPU6050 外观

MPU6050 模块通过 I2C 与 STM32F103 主控模块通信，由于该模块的 I2C 通讯引脚 SDA 及 SCL 已经连接了上拉电阻，因此它与外部 12C 通讯主机通讯时直接使用导线连接起来即可。

AD0 是 MPU6050 的从机地址设置引脚，当它接地或悬空时, 地址为: 0x68，当它接 VCC 时，地址为:0x69。

INT 是中断输出引脚，通过 MPU6050 的寄存器设置，可以让该模块在更新某一项数据后通过该引脚触发一次中断，让主控模块执行相应更新操作。经过前期测试，该中断不太稳定，最终采用了主控定时中断的读取 MPU6050 中的数据。

3.5 蓝牙通信模块

为了实现平衡小车和手机 APP 之间的通信，小车需要使用蓝牙模块。本项目选用了 JDY-31 蓝牙 3.0 模块，该模块是一款工作频段 2.4GHz，最大发射功率 8db，最大发射距离 30 米的经典蓝牙模块。

当模块上电而未连接时，模块进入 AT 指令模式，红色信号灯闪烁，此时我们可以直接通过 AT 指令配置蓝牙的名称、PIN 等。该模块只能作为蓝牙从机使用，但使用非常简单，当模块和手机已配对并建立连接后可进入透

传模式，此时红色信号灯长亮。

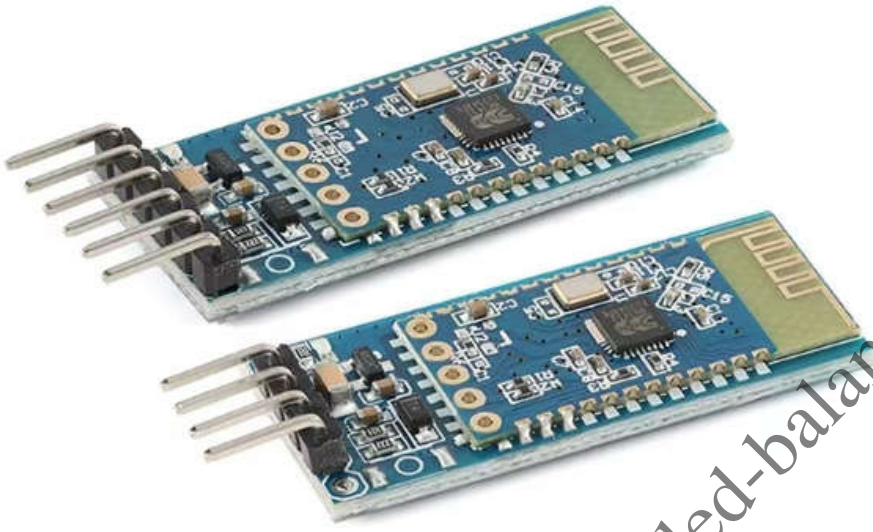

图 15 蓝牙串口模块

在透传模式下，主控通过 UART 向蓝牙模块收发数据，会在另一端蓝牙（比如手机、电脑的内置蓝牙模块）产生相应传输数据，我们只需使用 HAL 库提供的 UART 函数以及 Android 提供的蓝牙接口即可方便快捷地实现蓝牙通信功能。

3.6 WIFI 通信模块

WIFI 通信功能是平衡小车的拓展功能，通过 WIFI 小车可以接入互联网进行通信。使用 WIFI 通信的好处在于只要手机和平衡小车接入了同一网络，即可通过 TCP、UDP 协议远程传输数据。相较于蓝牙通信模块的近场无线通信，WIFI 通信具有更高的拓展性，可以实现更复杂的控制，比如平衡小车集群和多用户操作等。

本项目选用了 ATK-ESP8266 串口转 WIFI 模块，该模块由正点原子公司开发，学习资料丰富，提供了大量基础操作函数，可以大大提高我们的开发效率。

图 16 ATK-ESP8266 串口转 WIFI 模块

使用 WIFI 模块前，我们需要学习一些关于计算机网络的知识，特别是 TCP 协议的 IP、端口和报文结构等。在项目开发过程中，我们可以在微软商店下载 TCP UDP 网络调试工具进行开发和调试。

图 17 Windows TCP UDP 网络调试工具

通过创建 TCP Server，我们可以将个人电脑模拟成互联网，打开电脑的 WIFI，让手机和 WIFI 通信模块连接该 WIFI，两者便处于同一局域网，并被分配的不同的 IP，之后，WIFI 模块通过 AT 指令发起 TCP 连接，从而实现手机 APP 和平衡小车的互联和通信。

在主控模块中，我们可以定义宏变量，通过程序的初始化完成 WIFI 的自动连接，其中相关的配置包括电脑 WIFI 热点的名称 WIFI_SSID、密码 WIFI_PWD 以及手机在该网络下的 IP 和通信端口。

```
/*连接信息*/
#define WIFI_SSID           "mycloud"
#define WIFI_PWD            "12341234"
#define TCP_SERVER_IP       "192.168.137.1"
#define TCP_SERVER_PORT     "8025"
```

图 18 WIFI 通信配置示例

3.7 超声波测距模块

为了实现自动避障功能，本项目需要用到超声波测距模块，以便小车能感知前方障碍物的距离，及时做出回避运动。

本项目选用了新版 HC-SR04 超声波测距模块，具有高性能、宽电压、低价格等优点的工业级应用模块，测试范围可达 6 米，足以满足平衡小车的功能需求。

图 19 HC-SR04 超声波测距模块

该模块运用简单，只有两个信号端口。其中，Trig 端接收外部的触发新型号，当此管脚输入一个 10uS 以上的高电平时，可触发模块测距，Echo 端输出高电平信号，表示超声波往返时间的脉宽，可通过 STM32F103 芯片的 TIM 捕获功能直接接收和计算。

3.8 红外对管模块

为了实现小车的白底黑线循迹功能，我们可以根据黑色、白色对光线的吸收能力的差异选用红外线感知模块实现迹线的感知。

本项目直接采用现成的红外对管传感器实现小车对黑白颜色的感知。该模块具有一堆红外线发射管与接收管，可通过电位器旋钮调节检测距离，有效距离范围 2~5cm，探测角度 35%。

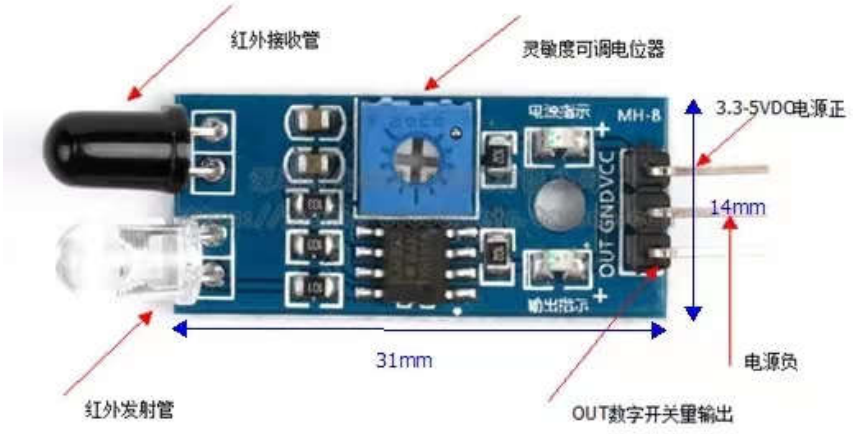

图 20 红外对管传感器

该模块只有一个信号输出端口，当红外对管对准白底时，红外发射管发出的红外线被反射至接收管端，绿色指示灯亮起，out 端输出低电平信号；当红外对管对准黑线时，红外发射管发出的红外线被吸收，红外接收管无法接收反射信号，指示灯熄灭，out 端输出高电平信号。

4 电路设计

4.1 电源设计

经过前面的模块介绍和选择可以知道小车需要一个能提供 7.4V 额定电压以及 5.0V、3.3V 稳定电压的电源来为电机和各电子模块提供电力。由于小车需要独立运行，所以本项目选择在车身固定锂电池组供电，以减轻小车负载和避免外部电源接线对小车平衡的干扰。

由于小车的电机采用 TB6612FNG 驱动，我们可以选用 12V 的锂电池组作为主电源，直接接入 TB6612FNG 的 VM 端，通过 PWM 控制电机驱动电压。

图 21 直流 12V 锂电池组

接着我们需要将 12V 主电源通过稳压模块得到 5.0V 和 3.3V 的 VCC 输出以供其他电子模块使用。

本项目采用 LM2576 系列 3A 降压 DC-DC 稳压器芯片，足以满足跟模块的电源需求情况。该芯片的经典应用电路如下：

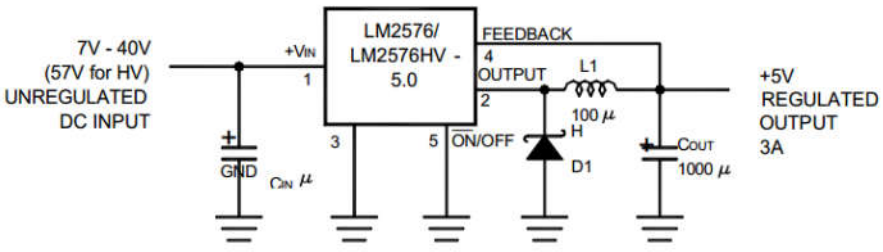

图 22 LM2576 DC 稳压器芯片经典应用电路

由于最终 PCB 会通过铜柱螺丝固定在电池组上方，为了方便引脚的焊接本项目采用了 TO263-5L 贴片封装的稳压芯片。

4.2 原理图设计

经过粗略估计，平衡小车的 PCB 大小可以限制在 10cm*10cm 的范围内，该设计满足嘉立创为用户提供每月两次免费打印服务，所以本项目的电路设计采用嘉立创 EDA 进行原理图设计和 PCB 制作。

在嘉立创 EDA 创建平衡小车工程，根据功能我们可以将电路划分为：电源模块、主控模块、蓝牙模块、WIFI 模块、陀螺仪模块、超声波模块、红外模块和电机驱动模块，具体原理图详见附录-原理图设计。

图 23 原理图目录

4.3 PCB 设计

绘制并检查好原理图后，我们可以使用嘉立创 EDA 的原理图转 PCB 功能进行 PCB 板的绘制。PCB 的绘制需要考虑模块和元件的位置，板材的尺寸位置以及走线等因素，以免因为不合理的设计而重复打板。

在板材尺寸方面，为了能让 PCB 固定在车轮底盘上，我们首先要根据底盘设计图纸提供的尺寸，在 PCB 上挖出四个螺丝孔。

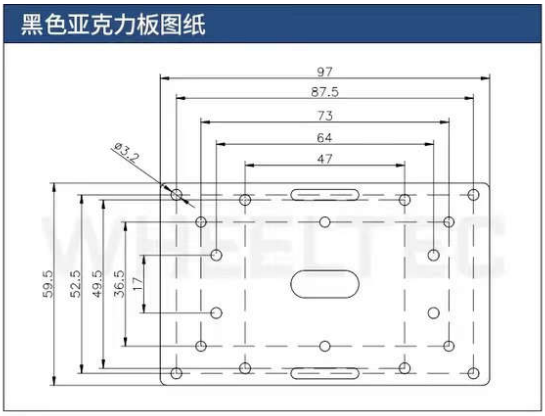

图 24 小车底板设计图纸

在元件摆放方面，特别要注意 MPU6050 的摆放方向，这会直接影响到姿态坐标系方向。超声波测距和循迹主要用于车前进方向，所以本设计将其摆放在正前方，电源芯片、电机接线插座等成对出现的元件则尽量对称摆放，此外还需要注意将接口类型的元件放在边缘等。

最终设计的 PCB 线路如下所示：

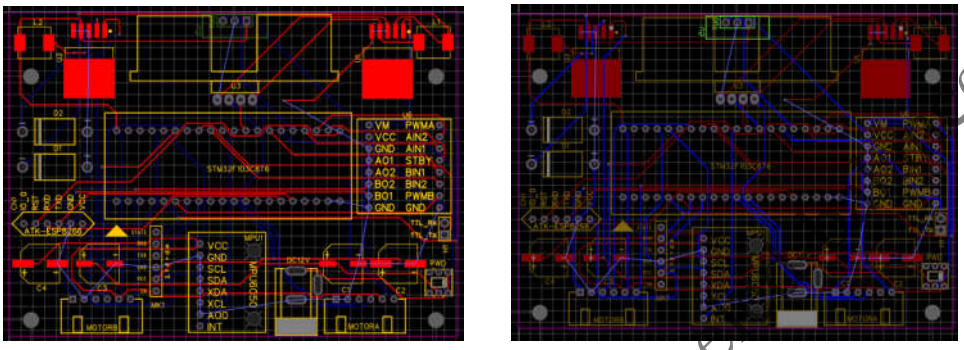

图 25 PCB 布局与连线

最后通过铺铜完成 GND 线路的连接。

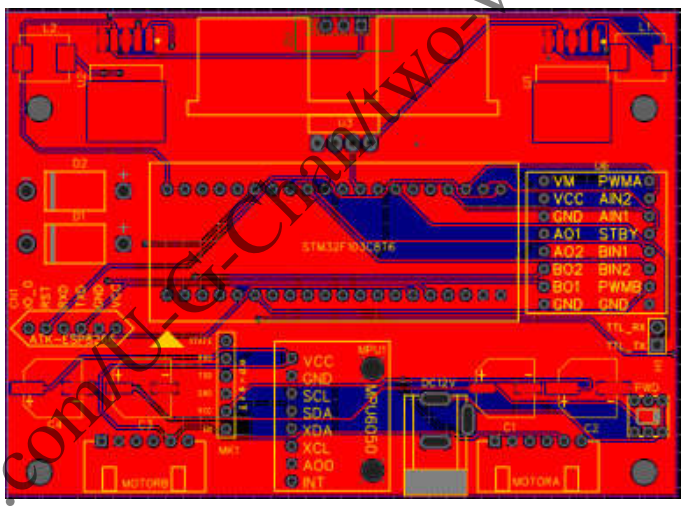

图 26 PCB 设计

4.4 制板与焊接

绘制好 PCB 设计图后，进行最后的检查，就可以上传 PCB 文件到“嘉立创下单助手”进行 PCB 打样了。最终 PCB 如下所示：

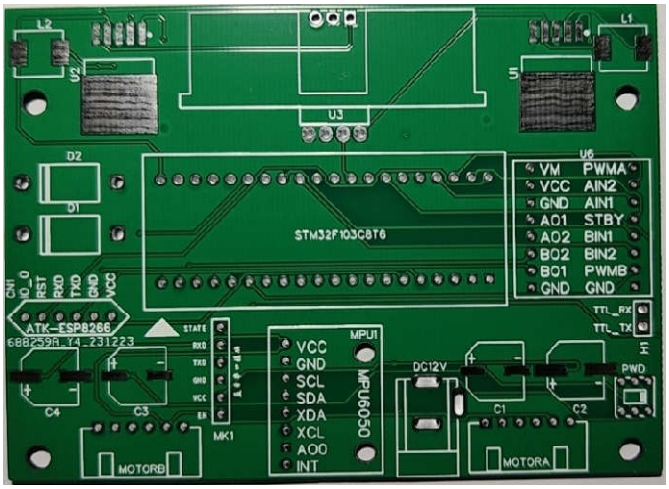

图 27 PCB 实体

最后将元件焊接上，插上各个电子模块，便得到最终成品。

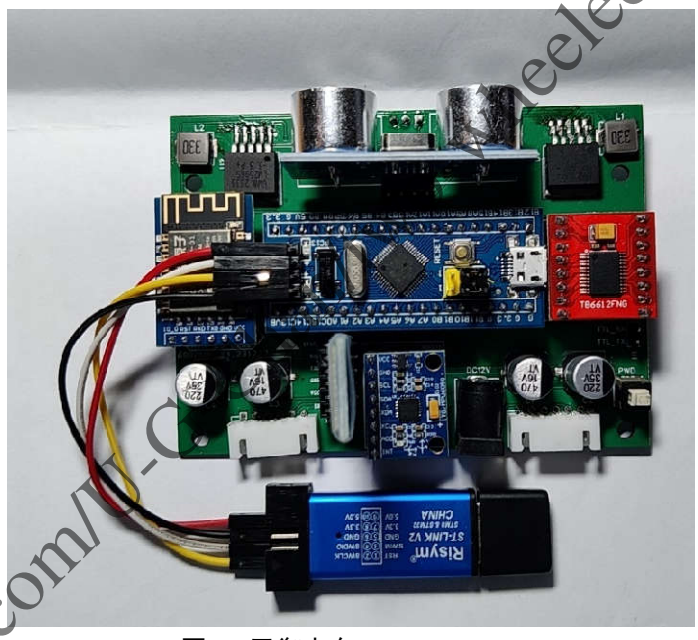

图 28 平衡小车 PCB

5 小车程序开发

5.1 程序主框架

本项目的单片机开发平台为：

- Keil_v5 IDE 软件
- STM32CubeMX 工程管理软件

本项目的程序主框架如下：

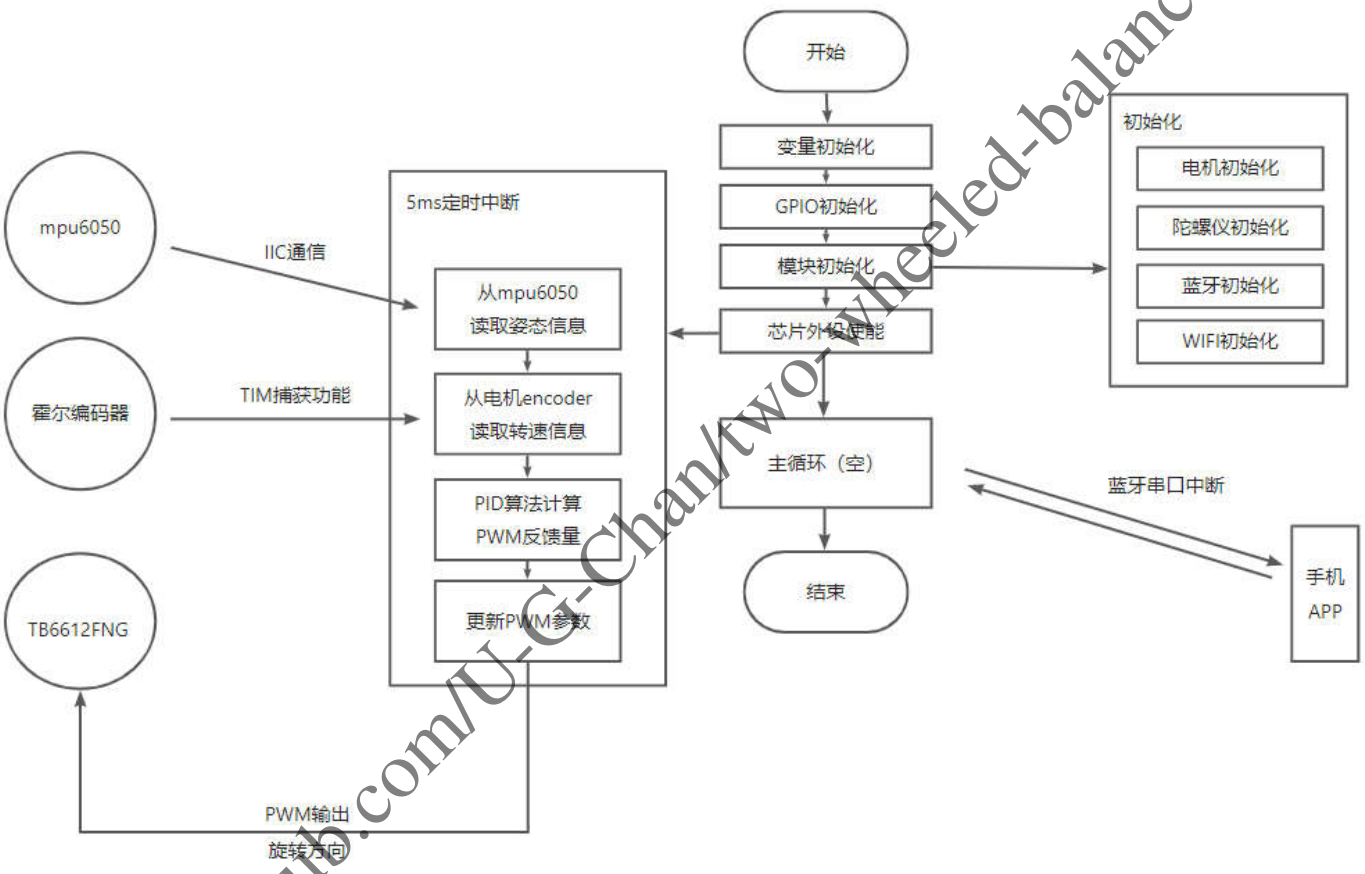

图 29 智能平衡车程序框架

本项目基于 STM32CubeMX 生成的工程代码开发。在 main 的主循环中，不进行任何操作。在所有初始化操作完成后，通过 TIM1 设置 5ms 的溢出中断，定期执行控制函数，完成小车的基本运动功能。

小车运动控制函数包括 4 大部分，分别是读取 mpu6050 中的姿态信息，读取当前捕获的霍尔编码器计数值，根据小车的姿态和运动信息计算 PID 控制反馈量，以及根据反馈量控制电机转动。

此外，小车通过串口中断实现蓝牙模块和手机之间的通信。两者的

通信以特定的数据包进行，格式为：

$$\{\{msgCode, arg1, arg2, \ldots\}\}$$

其中的$msgCode$是信息帧的编码，供程序识别并执行相应的控制函数，$argn$表示附加变量，经过字符串转化成控制函数的输入参数。

5.2 引脚分配与初始化

引脚分配应该与电路设计章节中的主控模块对应，在 STM32CubeMX 中的引脚配置如下图所示。

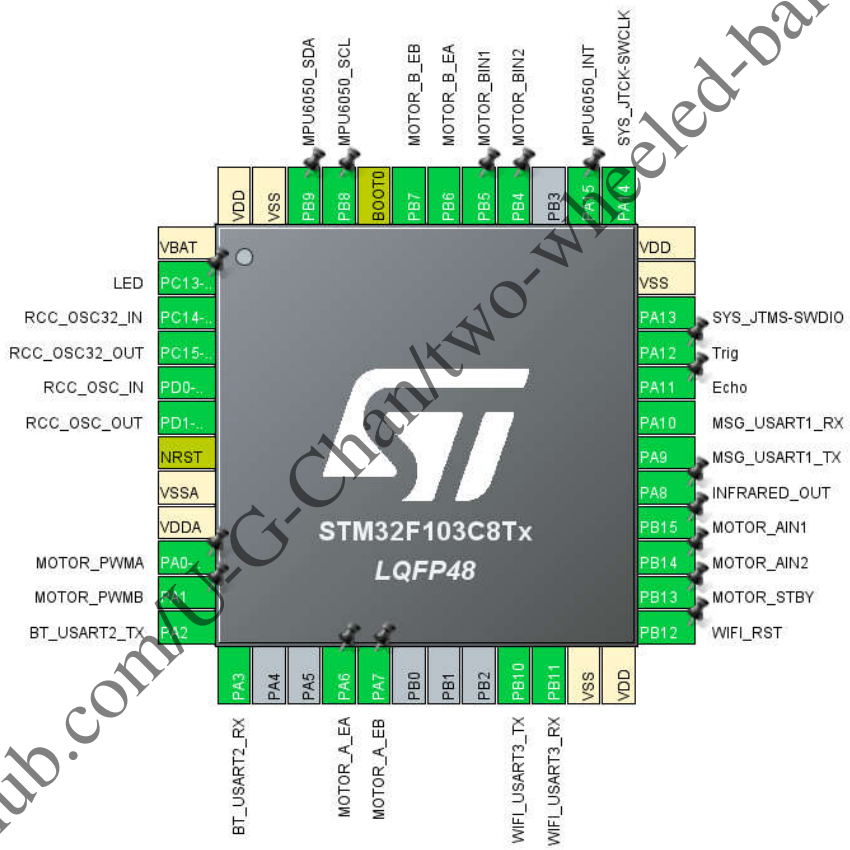

图 30 引脚分配

对于串口引脚，USART1(PA9,PA10)直接引至排针，以供 USB 调试；USART2(PA2,PA3)分配给蓝牙通信模块；USART3(PB10,PA11)分配给 WIFI 通信模块。

对于 IIC 引脚，I2C1(PB8,PB9)分配给 MPU6050 通信。

TB6612FNG 的方向控制引脚分别与 PB4、PB5，PA4、PA5 连接，PWMA、PWMB 由 PA0、PA1 产生，霍尔编码器分别与 TIM3、TIM4 的编码器模式对应的引脚相连。

PA11、PA12 则分配给超声波模块的 Echo、Trig 引脚，其中 PA11 设置为输入捕获模式。

5.3 主要控制函数介绍

回调函数：包括蓝牙、WIFI 的串口中断函数，超声波测距的捕获回调函数，超声波测距计数溢出更新回调函数（用于计算时间）。

```c
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim) {
  if (htim->Instance == TIM1) {
        if(mpu_init_flag == HAL_OK){
            control_pwm();
            ultrasonic_tim_overflow();
        }
  }
}

void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)  {

      if (huart->Instance == USART2)  //当前接收中断UART2
    {
            //printf("%c\r\n",rx_data);
            bluetooth_fm_buf_it(&rx_data);

    }
}

void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim){
    if (htim->Instance == TIM1){
        if (htim->Channel == HAL_TIM_ACTIVE_CHANNEL_4){
            ultrasonic_echo(htim, TIM_CHANNEL_4);
        }
    }
}
```

图 31 主要代码-回调函数

接收信息帧函数：检查帧的格式“{{}}”，提取其中的数据。

```c
/**
中断接收信息帧并处理
*/
void bluetooth_fm_buf_it(uint8_t* fm_data){
    //printf("%c\r\n",*fm_data);
    if(bluetooth_fm_idx < FMBUFFERSIZE){
        if(*fm_data == '{' && frame_receive_flag < 2){
            frame_receive_flag++;
            HAL_UART_Receive_IT(&HUART_BLUETOOTH,fm_data,1);
            return;
        }
        if(frame_receive_flag == 2){
            bluetooth_buffer_fm[bluetooth_fm_idx++] = *fm_data;
        }
        if(*fm_data == '}' && frame_receive_flag >= 2){
            frame_receive_flag++;
            if(frame_receive_flag >= 4){
                bluetooth_msg_solve_frame((char*)bluetooth_buffer_fm);
                bluetooth_fm_buf_clear();
                frame_receive_flag = 0;
            }
        }
    }
    if(bluetooth_fm_idx >= FMBUFFERSIZE)
    {
        bluetooth_fm_buf_clear();
    }
    HAL_UART_Receive_IT(&HUART_BLUETOOTH,fm_data,1);
}
```

图 32 主要代码-信息帧处理函数

APP 操作函数：根据手机 APP 传入的信息，执行相应的响应函数。

```c
void bluetooth_msg_solve_frame(char frame[]){
    char *token = strtok(frame, delimiter);

    int msgCode = 0;
    float args[ARG_LIMIT];
    uint8_t arg_idx = 0;

    while(token != NULL){
        args[arg_idx++] = atof(token);
        token = strtok(NULL, delimiter);
        if(arg_idx >= ARG_LIMIT){
            break;
        }
    }
    msgCode = (int)args[0];
    //根据信息码执行响应函数
    switch (msgCode){
        case 121:{led_test(args[1]);break;}
        case 101:{control_set_pid(PID_STAND,args[1],args[2],args[3]);break;}
        case 102:{control_set_pid(PID_VELOCITY,args[1],args[2],args[3]);break;}
        case 103:{control_set_pid(PID_TURN,args[1],args[2],args[3]);break;}
        case 300:{control_key(KEY_RLS,args[1]);break;
        case 301:{control_key(KEY_UP,args[1]);break;}
        case 302:{control_key(KEY_DOWN,args[1]);break;}
        case 303:{control_key(KEY_LEFT,args[1]);break;}
        case 304:{control_key(KEY_RIGHT,args[1]);break;}
        case 401:   {mpu6050_set_listen_state();break;}
        case 402:   {ultrasonic_set_listen_distance();break;}
        case 403:   {control_set_listen_setting();break;}

        default:break;
    }
}
```

图 33 主要代码-控制相应函数

运动控制函数：实现小车的平衡、运动等功能。

```
//直立控制PWM
int control_pwm_balance(float Angle, float Gyro){
    float Bias=Angle-ZERO;//求出平衡的角度中值 和机械相关
    int balance= Balance_Kp*Bias+Gyro*Balance_Kd;//计算平衡控制的电机PWM  PD控制    kp是P系数
    return -balance;
}
```

图 34 主要代码-直立控制函数

```
//速度控制PWM
int control_pwm_velocity(int encoder_left,int encoder_right){

    Encoder_Least =(encoder_left+encoder_right) - velocity_target;
    Encoder *= 0.8;
    Encoder += Encoder_Least*0.2;
    Encoder_Integral += Encoder;
    //Encoder_Integral=Encoder_Integral ;
    if(Encoder_Integral >  ENCODER_INTEGRAL_LIMIT)  Encoder_Integral =
        ENCODER_INTEGRAL_LIMIT;
    if(Encoder_Integral < -ENCODER_INTEGRAL_LIMIT)  Encoder_Integral =
        -ENCODER_INTEGRAL_LIMIT;
    Velocity=Encoder*Velocity_Kp+Encoder_Integral*Velocity_Ki;
    return Velocity;
}
```

图 35 主要代码-速度控制函数

```
//转向控制PWM
int control_pwm_turn(float Set_turn,float Gyro_Z)//转向控制
{
  int PWM_Out=0;
    if(Set_turn==0)
    {
     PWM_Out=Turn_Kd*Gyro_Z; //没有转向需求，Kd约束小车转向
    }
    if(Set_turn!=0)
    {
     PWM_Out=Turn_Kp*Set_turn; //有转向需求，Kp为期望小车转向
    }
    return PWM_Out;
}
```

图 36 主要代码-转向控制函数

6 手机 APP 开发

6.1 功能需求分析

为实现远程控制平衡小车，本项目需要开发一款手机 APP，以实现手机蓝牙和平衡小车蓝牙模块之间的相互通信。该 APP 的主要功能包括：

- 可以搜索周围的蓝牙设备，并显示在 APP 页面；
- 可以连接已配对的蓝牙设备，无线连接平衡小车；
- 可以将小车的运行时输出的日志信息显示在 APP 的终端面板；
- 可以切换小车的运行模式，实现人工控制、自动避障和黑线循迹等功能；
- 可以人工控制模式下，可以控制小车的前后左右移动；
- 可以实时监控小车的姿态角、前方障碍物和小车参数等信息；
- 可以设置小车的 PID 参数，方便平衡调试。

6.2 Android APP 开发基础

由于本人的手机系统基于 Android 系统内核，所以本项目的手机 APP 基于 Android 平台开发，以便实机调试和使用。

开发一款 Android APP 的基本步骤：

- 安装 Android Studio；
- 学习 Java 编程基础；
- 学习 MVVM 框架的概念和思想；
- 通过网络教学视频熟悉开发流程；
- 阅读 Android Studio 开发指南，熟悉常用组件；
- 根据模板构建 APP 页面；
- 设计页面布局和组件渲染；
- 为组件添加互动逻辑和功能函数；
- 实机调试和完善功能；
- 下载 APP 至手机。

6.3 页面设计与开发

本平衡车的手机 APP 采用底部导航的常见页面设计，共分为 Home、Dashboard 和 Notification 三大页面。

Home 页面由上端的信息终端，左下端的模型切换按钮和右下的方向控

制按钮组成，其外观如下图所示。

图 37 手机 APP-HOME 页面

上方的灰色方框是信息终端，小车运行的信息可以通过蓝牙串口发送到手机并显示，当信息过多时可以通过上下滑动查看最早或最新的信息记录，还可以通过“清除信息”按钮和“输出信息”开关控制信息输出情况。左下角的三个按钮用于模式切换，右下角的方向按钮仅在控制模式下有效。

Dashboard 页面是信息控制台，由上端的 PID 参数设置板块和下方的蓝牙连接板块组成，各个可以上下滑动查看更多内容，外观如下图所示。

图 38 手机 APP-DASHBOARD 页面

上半部分是直立换、速度环和转向环的 PID 参数设置表单，在输入框中输入数值后点击更新按钮，平衡小车会通过蓝牙模块收到数据包并执行更新函数若表格为空对应的参数将被设置为 0。下半部分是蓝牙搜索和连接板块，其中“balancecar2wheels”就是平衡小车的蓝牙模块的名称，在手机蓝牙打开的情况下，点击对应蓝牙设备即可实现蓝牙连接。

Notification 页面是信息监控台，分为三块数据监听对象，其页面外观如下图所示。

图 39 手机 APP-NOTIFICATION 页面

点击“监听姿态”按钮后，手机 APP 通知让小车实时发送姿态数据信息，以便用户检验小车 mpu6050 提供的数据是否有误。点击“监听测距”按钮后，小车会实时返回前方障碍物到小车的距离。最下则展示小车的其他参数设置情况，每点击一次“刷新参数”按钮获取一次。

完成小车的页面设计后，即可开发 APP 的交互内容。首先需要为每个页面创建相应的 Fragment 类，实现页面从创建到销毁的生命周期控制；其次为每个

页面创建相应的 ViewModel 类，实现后端数据的绑定操作；最后编写 APP 的蓝牙基础操作服务类。

整个 APP 工程源代码以电子版附录的形式提供，其主目录结构如下图所示。

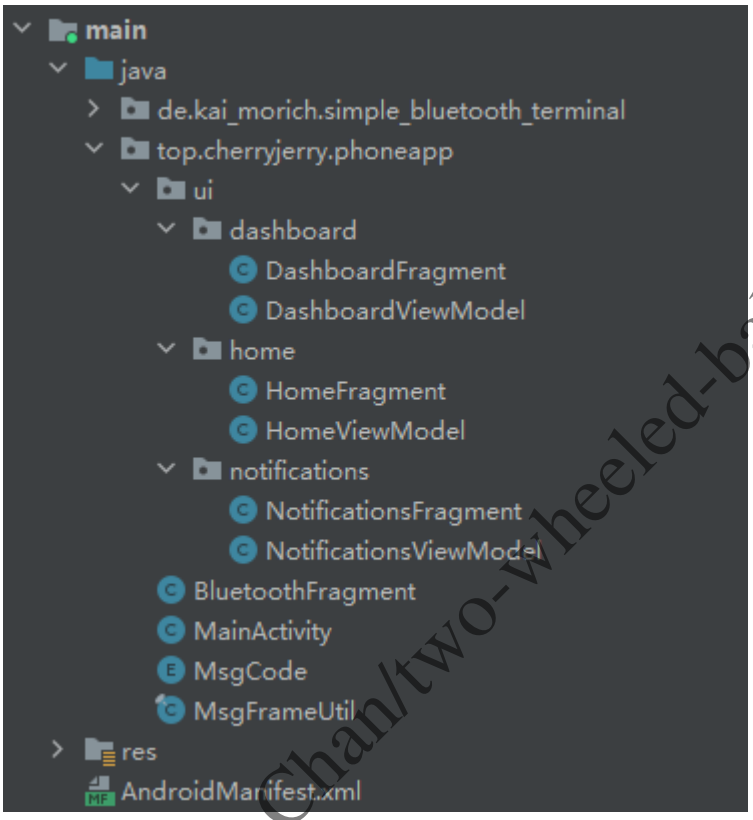

图 40 手机 APP-工程目录

7 PID 参数调试

7.1 参数极性测试

由于传感器的安装方向和设置不同，以及 PID 公式的编写原因，反馈值的正负会有所不同，因此在参数调试之前需要先确定好参数的极性。为了防止调试过程中各个环路之间的影响，需要屏蔽其他不调试环路。

首先，确定直立环 PD 参数的极性。将直立环、速度和转向环的参数置零，再将直立环的K_p参数分别改为正或负值，倾倒小车，观察车轮的转动方向，如果车轮的转动方向和倾斜方向一致，比如小车向前倾倒时车轮的旋转带动小车前进，说明极性正确，否则将K_p取相反数。

其次，确定速度环 PI 参数的极性。将直立环、速度和转向环的参数置零，再将将速度环的K_p参数分别改为正或负值，进行测试。平衡小车调速使用的是正反馈，这是因为小车在以一定速度运动时要想让小车停下，正常逻辑是让小车减速，但在两轮平衡过程中，小车反而要加速以减小小车的倾斜程度，再通过直立环调节小车的平衡。因此，在确定速度环参数时，可轻微转动小车车轮，如果轮子不断加速至最大值，说明极性正确；如果转动轮子时感受到更大阻力，且另一个车轮反方向旋转，则说明极性错误。

最后，确定方向环的参数极性。将直立环、速度的参数置零，设置一个期望角速度值，用手转动小车，如果小车车轮在助力转动，说明极性正确; 如果小车车轮在阻碍转动，说明极性错误。此外，在不同正负的数值下从同一侧面观察车轮旋转，小车的两个轮子朝相反方向旋转。

7.2 直立环调试

在速度环和转向环的参数置零的条件下：

调试直立环K_p：将直立环K_d置零，再逐步增加K_p参数的大小，当小车基本可以稳定以后继续增大K_p，直到小车出现低频抖动。

调试直立环K_d：再上一步的基础上，逐渐增大K_d，直到小车出现高频抖动为止。

最后取直立环K_p、K_d数值为原来的 0.6 倍。

7.3 速度环调试

在保持上一步直立环参数，转向环参数置零的条件下：

调试速度环K_p：再逐步增加K_p参数的大小，使得小车可以很好的保持直立，且原地不动。

调试速度环K_i：根据调试经验，速度环K_i总是取$\frac{1}{200}\cdot K_p$。

7.4 转向环调试

在保持直立环参数并传入转角期望值的条件下：

调试转向环K_p：逐渐增加K_p参数的大小，使小车可以很好保持直立且可以按照期望反向和期望速度旋转。

调试转向环K_d：逐渐增加K_d参数的大小，使小车可以很好保持直立且最终可以稳定在原地。

8 功能实现与测试

8.1 通过 APP 遥控小车

操作步骤：

- 按下小车 reset 按键；
- 打开手机 APP；
- 在 Dashboard 页面中连接小车蓝牙；
- 在 Home 页面中点击“控制模式”按钮；
- 点击 Home 页面中的方向键，观察小车运动情况。

8.2 自主运行与避障

操作步骤：

- 在 Home 页面中点击“自由模式”按钮；
- 在 Home 页面中监控障碍物距离；
- 在小车前防止障碍物，观察小车运动情况。

8.3 白底黑线背景循迹

操作步骤：

- 在“控制模式”下，控制小车运动到白底黑线背景板内；
- 让小车前方的红外对管靠近黑线；
- 在 Home 页面中点击“循迹模式”按钮；
- 观察小车左右转动，寻找黑线；
- 观察小车沿着黑线轨迹运动。

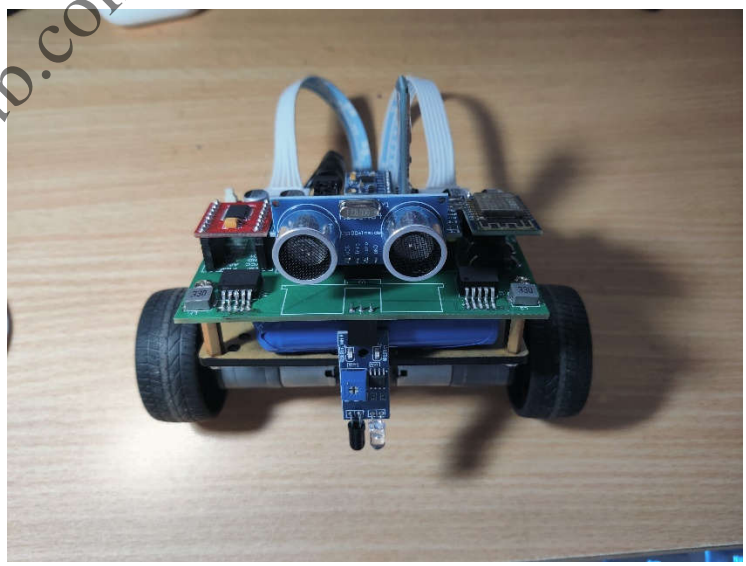

图 41 平衡小车实物-正面

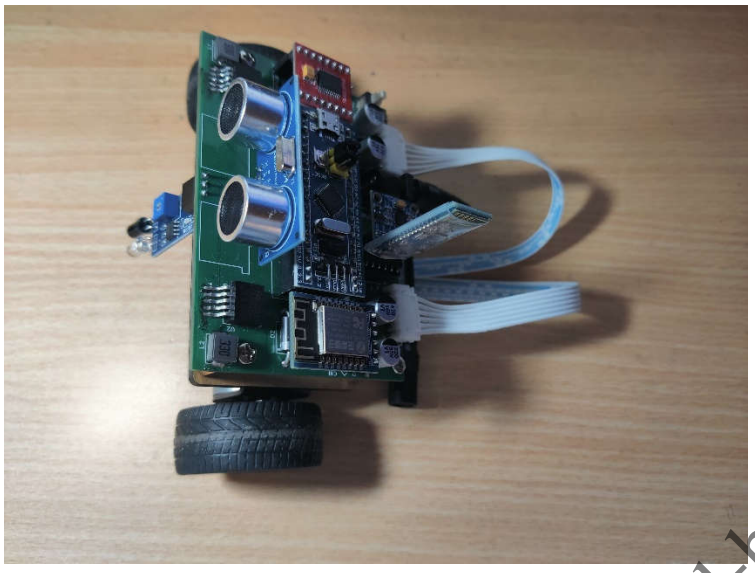

图 42 平衡小车实物-侧面

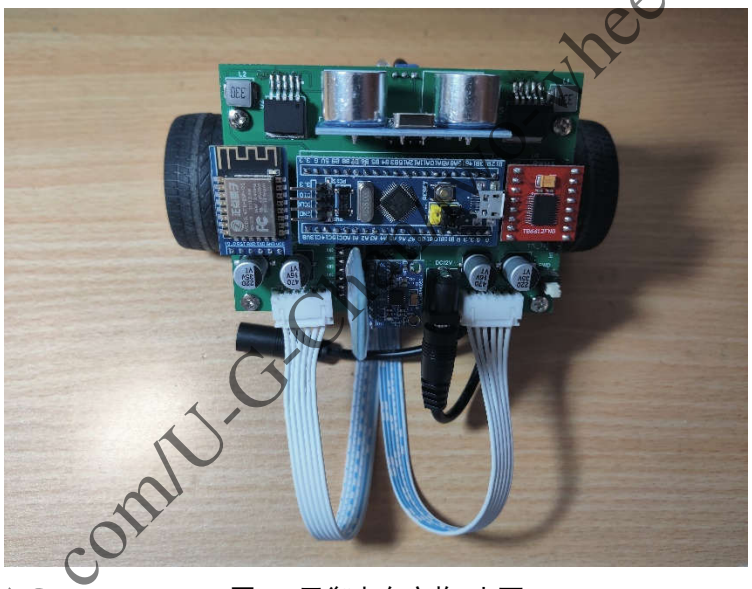

图 43 平衡小车实物-上面

9 总结

通过本次专业综合设计，我的电子工程素质和科学实验能力得到了巨大的进步。

在完成“智能平衡小车”项目的过程中，我自主学习单片机相关理论知识，查阅和整理有关参考资料，独立完成方案选择、电路设计、程序设计和报告撰写任务，最终制作出一款由 MPU6050 陀螺仪模块、TB6612FNG 直流驱动模块和两个直流电机作为运动机构，可以通过蓝牙或 WIFI 模块与手机通信，可以通过超声波测距模块和红外对管模块感知外部空间情况并自动运行和避障的平衡小车。

然而本次设计也存在一些不足之处，比如红外循迹采用了但对管方案，小车在循迹运动过程中会出现反复抖动、前进速度慢的问题，这点以后可以通过采用多红外对管阵列来提高循迹的准确度解决。又比如限于开发人数和开发时间，手机 APP 的 TCP 通信等功能没有来得及实现。尽管我的平衡小车并不完美，但每一个微小功能的实现都投入了我自己的巨大精力，从学习和实际的角度来说，整个综合设计过程和最终的作品都是十分具有意义和价值的。

本次综合设计是一次难忘的实践经历，我感受到了单片机技术在电子信息领域的广泛应用，学习并亲自动手实现自动控制原理中的经典 PID 算法，还学习了 Android APP 的开发，这些经历激发了我深入学习并综合运用专业技能解决项目问题的兴趣和热情，对我未来的学习和研究意义重大。

附录

1) 源代码-平衡小车（电子版）

2) 源代码-手机 APP（电子版）

3）嘉立创 EDA 工程文件（电子版）

4）电路原理图

5）PCB 印制板