机械与能源工程系
SUSTech Department of Mechanical and Energy Engineering

项 目 报 告

项目名称：两轮自平衡小车 PID 控制器设计和调试

课程名称：嵌入式系统与机器人

课程编号：ME432

目　录

一、项目目标

本项目使用现成的两轮平衡小车，从零开始调试陀螺仪、编码器、电机、蓝牙模块等各个模块并搭建电路，再对整个系统进行 PID 控制，首先使小车实现直立平衡的效果，并能够承受一定的扰动，再使小车实现以给定的速度前进或后退以及以给定角速度在水平面上旋转的效果。

通过增加蓝牙通信的功能，使得 STM32 开发板能够和手机进行通信，从而实现通过手机上 APP 发出的指令对小车的控制模式、期望速度、期望旋转角度等参数进行配置的功能。

二、基本原理和实现思路

本节将从平衡小车的机械结构、电路逻辑、控制器设计三个方面阐释本项目的基本原理和实现思路。

1）机械结构

如图 1 所示，小车的结构主要分为三层，三层分别包含了不同的机械结构，搭载了不同的电子器件。

最上层包含了系统电路的主要部分。该部分以亚克力板为载体，其上搭载了以面包板为载体的核心电路。核心电路连接 STM32 核心主控板、MPU9250 模块、驱动模块三大部分。除此之外，蓝牙模块也位于该层，通过杜邦线与面包板直接相连。

中间层以亚克力板为载体，由最上层亚克力板以及起支撑作用的四根铜柱构成一个相对独立的空间。该空间内搭载电池、分电板和 12V 转 5V 降压模块。

最下层包含分别驱动左右轮的两个电机以及搭载于其上的编码器。左右两个电机分别与左右两轮直接相连以驱动左右两个电机。

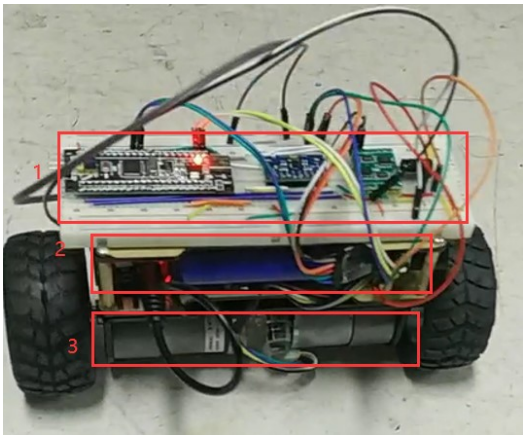

图 1 平衡小车实物图及其分层示意图

2）电路逻辑

电路逻辑主要分为供电逻辑部分和控制信号逻辑部分。供电逻辑部分相对次要：只需要将供电电路部分划分 5V 和 GND 的区域，并将各电子器件与和其所需电压相匹配的供电电路部分相连即可。相比之下，控制信号电路逻辑显得更为重要和复杂。故此处仅展现和阐释控制信号部分的电路逻辑。

平衡小车的控制电路逻辑图如图 2 所示。

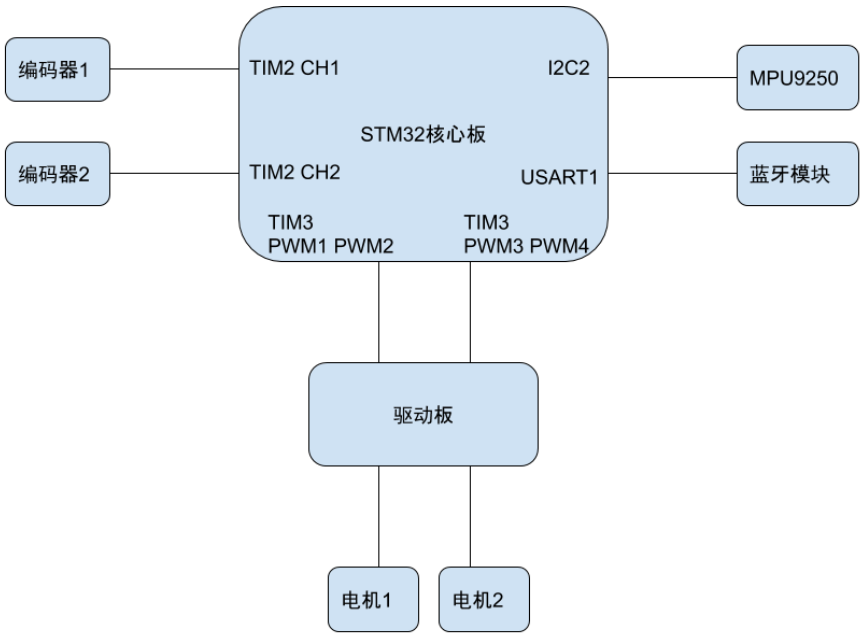

图 2 平衡小车控制电路逻辑框图

STM32 核心板是整个控制电路系统的核心部分。核心板分别和电机、编码器、MPU9250、蓝牙模块、驱动板等周边器件直接或间接相连。STM32 核心板芯片为 STM32F103C8T6，其带有多个功能各异的外设，可用于驱动不同的设备。

其中 TIM2 的 CH1 和 CH2 分别驱动编码器 1 和编码器 2；TIM3 的 PWM1 和 PWM2 共同控制电机 1 的转向和转速(通过 PWM1 和 PWM2 的占空比之差的极性控制电机 1 的转向，转速之差的绝对值控制电机 1 的转速，电机 2 的转速和转向控制同理)；通过 TIM3 的 PWM3 和 PWM4 共同控制电机 2 的转向和转速。驱动板作为 STM32 主控板和电机之间的媒介，将对 PWM 占空比之差的控制转换为对电机的输入电压的控制。

I2C2 用于 STM32 主控板和 MPU9250 陀螺仪模块间的数据通信。MPU9250 将陀螺仪、加速度计、磁力计各部分的测量数据通过 I2C 通信传输给 STM32 主控，由 STM32 主控板进行滤波和姿态解算等处理。

USART1 用于 STM32 主控板和蓝牙模块间的数据通信。蓝牙模块接收手机 APP 向其发出的指令和数据，这些指令和数据通过串口通信传输给主控板，从而使主控板根据其作出相应的处理。主控板也通过串口通信将数据发送给蓝牙模块，蓝牙模块将从主控板收到的数据传输给手机，并在手机 APP 上显示。

3）控制器设计

PID 控制分为直立 PID 控制、速度 PID 控制和转向 PID 控制。

直立 PID 控制指控制小车在具有一定扰动(如水平方向推动)的情况下保持直立，即保持通过小车两轮轮轴且垂直于小车各层的面(以下简称中轴面)和竖直方向的夹角为 0；速度 PID 控制指控制小车速度为一期望值。转向 PID 控制指控制小车 z 轴角速度为一期望值。

直立 PID 控制器的示意图如图 3 所示。

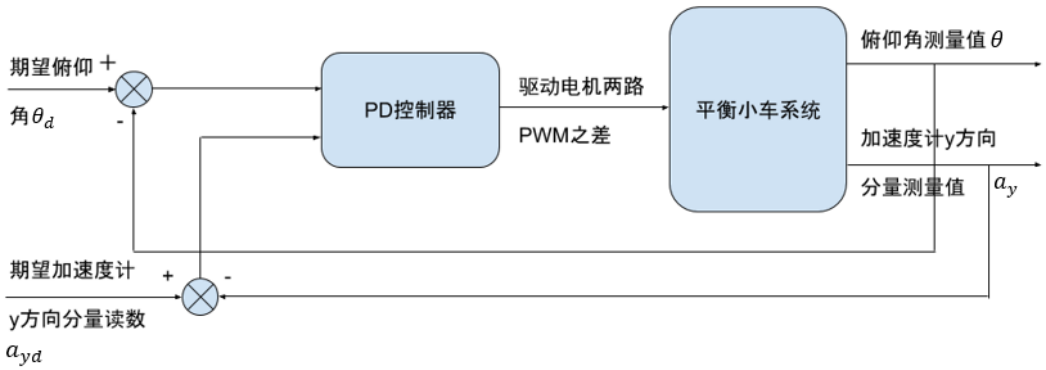

图 3 平衡小车直立控制器示意图

速度控制 PID 和转向控制 PID 建立在直立控制 PID 的基础之上。加入速度和转向 PID 后的控制器示意图如图 4 所示。

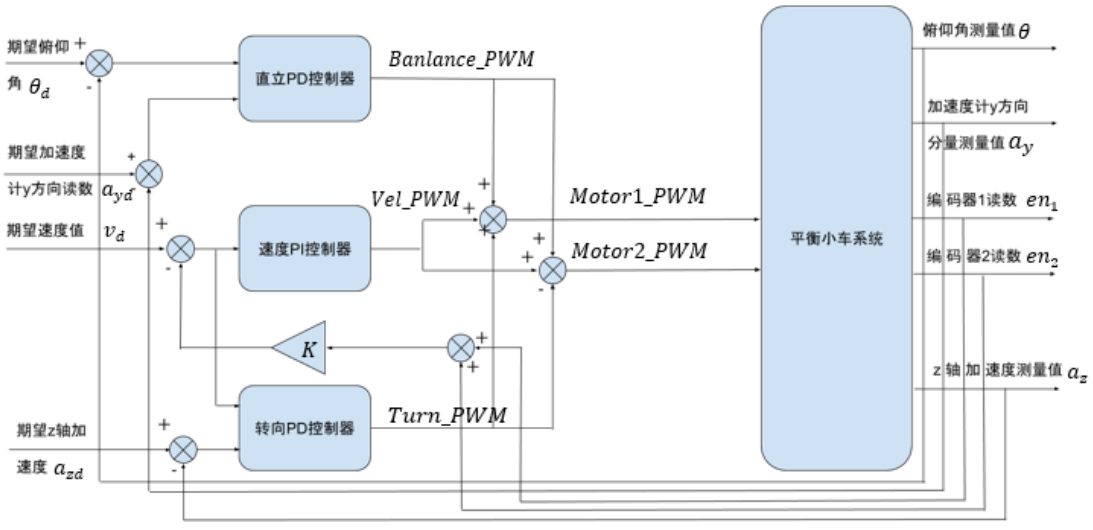

图 4 平衡小车直立控制器、速度控制器、转向控制器示意图

系统的输入为期望俯仰角θ_d，期望加速度计 y 方向分量读数a_{yd},期望 z 轴加速度a_{zd}。值得注意的是，期望速度v_d通过串级 PID 控制器算出。串级 PID 控制器为本项目中一项相对特殊的设计，上图未展示，将在第四部分详细阐释。输出为俯仰角测量值θ,加速度计 y 方向分量测量值a_y，编码器 1 的读数en_1,编码器 2

的读数en_2， z 轴加速度测量值a_z。输出经过一系列处理和运算后和输入作差，再经过一系列运算分别得到电机 1 和电机 2 的 PWM 之差$Motor1_PWM$和$Motor2_PWM$.。具体运算方式如下：

$$Banlance\_PWM = K_{p1}(\theta_d - \theta) + K_{d1}(a_{yd} - a_y)$$

$$Vel\_PWM = K_{p2}(v_d - v) + K_{i2}\int (v_d - v)\,dt$$

$$Turn\_PWM = K_{p3}(v_d - v) + K_{d3}(a_{zd} - a_z)$$

$$v = K(en_1 + en_2), K = \frac{f}{2 \times 4 \times 448}, f = 100Hz.$$

$$Motor1\_PWM = Banlance\_PWM + Vel\_PWM + Turn\_PWM$$

$$Motor2\_PWM = Banlance\_PWM + Vel\_PWM - Turn\_PWM$$

三、实现方法

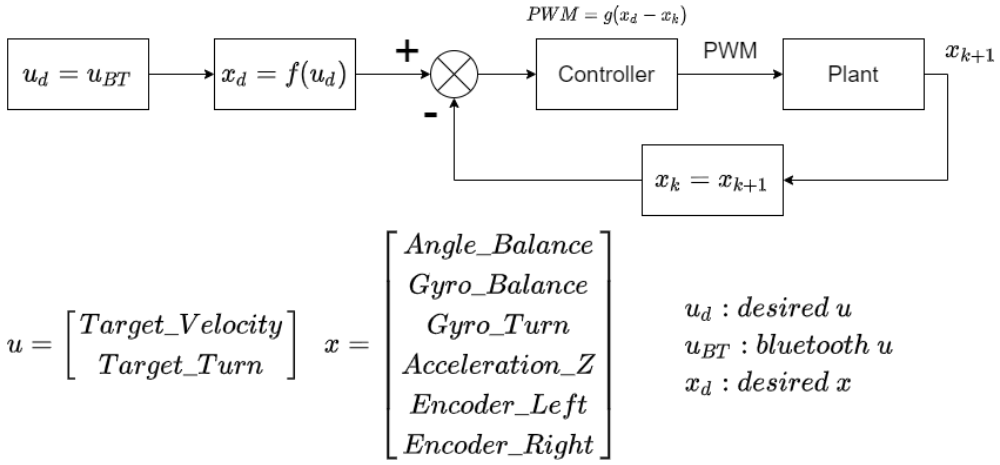

图 5 控制系统逻辑框图

本项目共实现直立平衡控制、车身速度控制、转向控制和蓝牙控制方式。

如图 7 所示，令u表示系统输入指令（目标速度和目标转向角度），x_k表示平衡小车在k时刻的系统状态，当平衡车完成初始化后即进入该控制循环。蓝牙通过串口中断输入指令u_{BT}，在$main$函数中在k时刻将输入指令转化为目标系统状态，并通过 IMU 和左右轮电机编码器读数解算当前系统状态。控制系统将目标系统状态和当前系统状态做差，得到状态误差作为 PID 反馈控制器的输入，并分别通过 PD、PI 和 P 控制器解算平衡、速度和转向控制所需的 PWM 值，经过限幅后作为控制量输入电机。进入$k+1$时刻后根据传感器读数更新系统状态$x_k = x_{k+1}$。

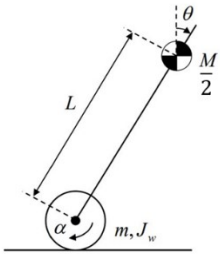

图 6 轮式倒立摆模型

小车的平衡控制通过 PD 控制器实现。如图 8 所示，平衡小车可以被近似为一个一阶倒立摆模型，为使小车实现站立，在理想条件下只有当小车与地面垂直时，车体重力方向与车轮支持力的方向正好在一条直线上，小车此时受力平衡，可以实现垂直站立。但是在现实环境中，θ 角度不总为零。如图 9 所示，当合力不为零时，根据牛顿运动定律，小车将向受力方向运动，导致 θ 角度越来越大，直到倒地。

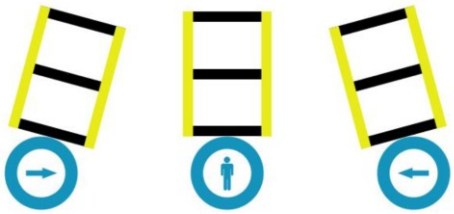

图 7 小车平衡和左右倒地示意图

根据图 9 所示的平衡小车的三种状态，我们把小车偏离平衡位置的角度作为偏差，我们的目标是通过负反馈控制，让这个偏差接近于零。理想状态下，只要我们可以控制电机的加速度和小车的倾角成正比，就可以让小车保持平衡。

$$PWM = K_p\theta$$

实际上，小车到达平衡的位置之后，因为 θ 为零，所以输出 PWM 为零，但刚体绕轴转动时具有的惯性，小车会往另外一个方向倒去，如此反复，小车会在平衡位置出现震荡而无法静止。为了让小车能静止在平衡位置附近，我们不仅需要在电机上施加和倾角成正比的回复力，还需要增加和角速度成正比而与运动方向相反的阻尼力，即引入微分控制。

$$PWM = K_p\theta + K_d\dot{\theta}$$

最终的直立 PD 控制代码如下。

```
int balance(float Angle, float Gyro)
{
    static int Bias, balance;
    // Compute pitch angle bias
    Bias = Angle - ZHONGZHI;
    // Compute PWM
    balance = Balance_Kp * Bias + Gyro * Balance_Kd;
    return balance;
}
```

小车的速度控制通过 PI 控制器实现。速度控制与平衡控制相比更为复杂，因为小车需要在进行直立控制的同时完成速度控制。因此，我们设计了一个串级 PID 控制器，该控制器具有一个外环来控制小车速度和一个外环来控制小车直立。所谓级联控制是指使用两个控制器串联工作，外环控制器的输出作为内环控制器的参考值，而内环控制器的控制输出作为控制系统的最终输出。

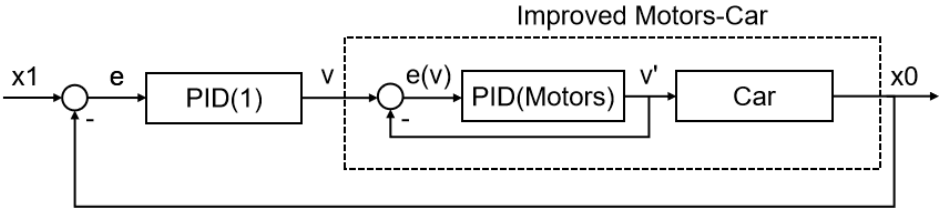

图 8 串级 PID 控制器程序逻辑框图

$$PWM_{velocity} = K_{p1} \cdot v + K_i \cdot \Sigma v$$

$$PWM_{velocity} = K_{p1} \cdot v + K_i \cdot \sum v$$

$$PWM_{balance} = K_{p2}(\theta - PWM_{velocity}) + K_d\dot{\theta}$$

之所以选择 PD 控制器作为内环控制器，而选择 PI 控制器作为外环控制器，是因为内环的平衡控制需要更高的控制频率，且微分项用于增加小车的回正阻尼。而外环使用 PI 控制器是为防止编码器噪声被放大并消除系统的静差。

考虑到 PID 参数整定，我们对串级 PID 公式进一步化简得到

$$PWM = K_{p2} \cdot \theta + K_d \cdot \dot{\theta} - K_{p2}\left(K_{p1} \cdot v + K_i \cdot \sum v\right)$$

可以看到该串级控制器可以拆分成两个独立的 PI 和 PD 控制器，即小车的速度和平衡控制器参数可以独立地调整。值得注意的是，独立施加的速度 PI 控制器是一个正反馈。

	霍尔测速	光电测速
电机电压	3-12V	3-12V
电机电流	250mA @12V	250mA @12V
额定功率	5W	5W
额定负载	3.0kg.cm	3.0kg.cm
堵转电流	3.5A	3.5A
堵转扭矩	10kg	10kg
空载转速	585rpm/min	585rpm/min
编码精度	16CPR	448CPR
减速比	1 : 18	1 : 18
编码器电压	3.3-5V	3.3-5V

图 9 电机和编码器的规格参数

由电机参数表可知光电编码器编码精度为 448CPR，即电机每转一圈产生 448 个方波。而实际采购的电机减速比为 1：10，因此编码器检测到 10 圈，输出轴旋转 1 圈。编码器配置为上下边沿检测，即编码器四倍频，故根据大 M 测速法，小车车身速度的计算公式如下

$$v = \frac{(encoder\_left + encoder\_right) * f * 60}{2 * 4 * 448}$$

其中$f = 100Hz$为编码器更新频率。

为了进一步消除编码器噪音，我们对更新的车身速度进行低通滤波，并对积分项限幅，防止误差产生累加效应，最终的速度控制代码如下。

```
1.  int velocity(int encoder_left, int encoder_right)
2.  {
3.      static int Velocity, Encoder_Least, Encoder;
4.      static int Encoder_Integral, Encoder_I_max;
5.      // Compute body velocity bias
6.      Encoder_Least = (encoder_left+encoder_right)/2*100*60/4/448 - Target_Ve-
    locity;
7.      Encoder *= 0.7;
8.      Encoder += Encoder_Least * 0.3;
9.      Encoder_Integral += Encoder;
10.     // Integral limit
11.     Encoder_I_max = 1000 / Velocity_Ki;
12.     if (Encoder_Integral > Encoder_I_max)
13.         Encoder_Integral = Encoder_I_max;
14.     if (Encoder_Integral < -Encoder_I_max)
15.         Encoder_Integral = -Encoder_I_max;
16.     // Compute PWM
17.     Velocity = Encoder * Velocity_Kp + Encoder_Integral * Velocity_Ki;
18.     return Velocity;
19. }
```

小车的转向控制使用的是独立的 P 控制器。因为小车的转向信息可以通过编码器或陀螺仪获取，所以小车的转向有多种控制方式，但是考虑到编码器信息无法反映车轮的滑动的问题，且小车的转向环对相应的要求不高，所以我们选择简单的角度 P 控制器。该控制器使用 IMU 获取的小车 z 轴转角，并与目标转向角做差得到转向偏差，并对该偏差进行比例控制。最终的控制代码如下。

```
1. int turn(int encoder_left, int encoder_right, float gyro)
2. {
3.     static int Turn, Bias;
4.     // Compute z-axis angular velocity bias
5.     Bias = Target_Turn-gyro;
6.     // Compute PWM
7.     Turn = Bias * Turn_Kp;
8.     return Turn;
9. }
```

小车通过蓝牙模块接收手机发送的控制指令，其指令格式为

$$f/b/l/r\ +\ val\ +\ A$$

其中第一个字母f/b/l/r分别代表前进/后退/左转/右转，val为对应控制的目标控制量，字母A为结束标识符。当核心板接收到如上指令后便会根据val的值修改对应的速度控制目标量$Target_Velocity$或转向控制目标量$Target_Turn$。

四、实验数据与结果分析

1）直立控制调参实验数据

从第四部分得知，平衡小车直立环使用 PD 控制器。

将平衡小车放在地面上，绕电机轴所在的直线旋转平衡小车，通过 Keil 的 debug 模式中 watch 的实时监控数据记录小车直立时中轴面和竖直方向间的夹角，记为参数 ZHONGZHI。接着确定K_p的极性。首先尝试K_p取负值。观察发现，当K_p为负时，小车加速倒下，和预期效果相反，故确定K_p极性为正。从 10 开始增大K_p 。起初K_p较小时，小车在失去外界辅助后立即向一个方向倒下，可见控制效果不明显。随着K_p不断增大，小车开始出现小车大幅度的震荡。此时需要增加微分控制以抑制大幅震荡。接着整定参数K_d 。首先确定K_d的极性。尝试K_d取正值，观察发现，K_d对大幅震荡丝毫没有抑制效果。再尝试K_d取负值，观察发现小车震荡幅度明显减小，故确定K_d的极性为负。不断增大K_d的绝对值，直到小车

能够平稳直立。最终我们确立的直立调试参数为K_p=70，K_d=-3，ZHONGZHI=6.5。

2）结果分析

将整定了的K_p、K_d 、ZHONGZHI 的程序烧录进开发板，观察发现，平衡小车几乎没有震荡和抖动，相当平稳，能够保持一段时间的站立。对小车施加小幅扰动，小车向一个方向倾倒，但很快可以恢复到原来的直立状态。但是，一旦扰动幅度稍大，小车向一个方向倾倒后便无法恢复直立，而是倾斜着向倾倒的方向运动。为改善这方面的控制效果，需要在直立控制的基础上加入速度控制。

速度控制使用 PI（比例积分）控制器，这也是速度控制中常用的控制器。P 控制器可增大系统的响应速度，从而使得平衡小车在受到扰动后迅速恢复平衡，但无法消除稳态误差，即期望速度和实际速度间的偏差无法消除。而 I 控制器的引入可以消除稳态误差，从而使得实际速度和期望速度吻合。实际调参过程有待后续研究。

五、参考文献

[1]李帅男.基于 STM32 控制的双轮自平衡小车的设计[J].现代工业经济和信息化，2018，8(13):34-35.

[2]杨志强.基于 STM32C8T6 的智能二轮自平衡小车的设计[J].电子测试，2020(17):9-11+37.

[3]李声福,周恒,马超.基于 STM32 的平衡小车控制系统设计[J].电子制作，2020(19):5-7.

[4]周冲，王振，陈贵茂.智能平衡小车与 PID 控制算法[J].电脑编程技巧与维护，2020(09):128-130.

[5]王秋生，赵聪. 基于 STM32 的双轮自平衡小车[A]. 全国地方机械工程学会、河南省机械工程学会（承办）.2019 年第九届全国地方机械工程学会学术年会论文集[C].全国地方机械工程学会、河南省机械工程学会（承办）:河南省机械工程学会，2019:5.