**自平衡小车的设计和实现**

时间：2020.11.06

作者： 楠 风

目 录

[1、背景 3](#_Toc59200025)

[2、总体设计 4](#_Toc59200026)

[3、算法设计 5](#_Toc59200027)

[3.1 卡尔曼滤波算法 5](#_Toc59200028)

[3.2 一阶互补滤波器 7](#_Toc59200029)

[3.3 一阶低通滤波器 7](#_Toc59200030)

[3.4 PID算法 7](#_Toc59200031)

[4、硬件设计 9](#_Toc59200032)

[4.1 姿态感知单元 9](#_Toc59200033)

[4.2 测距单元 10](#_Toc59200034)

[4.3 无线传输单元 11](#_Toc59200035)

[4.4 显示模块 12](#_Toc59200036)

[4.5 驱动模块 13](#_Toc59200037)

[4.6 电机模块 14](#_Toc59200038)

[4.7 核心系统 15](#_Toc59200039)

[5、软件设计 16](#_Toc59200040)

[5.1 通信模块 16](#_Toc59200041)

[5.2 自平衡模块 17](#_Toc59200042)

[5.3 显示模块 18](#_Toc59200043)

[5.4 电脑遥控 18](#_Toc59200044)

[6、总结 19](#_Toc59200045)

[6.1 初步执行计划 19](#_Toc59200046)

[6.2 经验总结 19](#_Toc59200047)

[7、参考文献 21](#_Toc59200048)

1、背景

因对电子设计和机器人的热爱，想在工作之余发展自己的爱好，故计划制作一个自平衡的小车机器人，后期还打算在上面搭载诸如智能语言、计算机视觉、人工智能等功能模块，目标很远大，但道阻且长，希望可戒贪、嗔、痴，顺利完成该项目。

2、总体设计

自平衡小车的设计主要包含两个部分，硬件设计和软件设计。硬件上使用STM32F103系列的芯片作为主控芯片，还涉及许多传感器和驱动模块，具体如下表所示。

表2.1 硬件设计

|  |  |
| --- | --- |
| 功能 | 模块 |
| 主控系统 | STM32F103C8T6 |
| 姿态感知 | MPU6050 |
| 无线数据传输 | 蓝牙4.0 |
| 测距功能（未使用） | 超声波测距 |
| 电机（300rpm） | 带码盘的减速直流电机N20 |
| 电机驱动 | TB6612FNG芯片 |
| 显示模块 | OLED |

介绍完硬件的总体设计后，下面给出软件设计。软件设计主要分为底层的硬件设备驱动设计，各个功能模块的设计和业务逻辑的设计。详细设计如图所示。



图1.1 软件架构

3、算法设计

考虑MPU6050传感器的加速度输出包含了很多噪声，而角速度输出存在零漂的情况，需要某种算法对数据进行处理和融合，卡尔曼滤波算法不仅实时性强，滤波性能好，而且常用于控制系统的滤波，因此考虑使用该算法作为传感器的滤波算法，除此外还介绍了一阶低通滤波器和一阶互补滤波器。而针对小车的自平衡算法设计，这里选用工业中最常用的PID控制算法，下面主要详细介绍了两类算的原理和针对平衡小车的具体应用。

3.1 卡尔曼滤波算法

首先引入一个离散的控制过程系统[2]，具体如下描述：





其中式为系统的过程模型，式为测量模型，和分别对应系统时刻的系统状态和系统控制量，和是系统参数，对于多系统来说可以是矩阵。是系统时刻的测量值，为测量系统参数，对于多系统该参数可以是矩阵。和为过程和测量噪声，经常假设为高斯白噪声进行处理，其协方差分别为和。

下面给出卡尔曼滤波的五个公式：

假设当前系统状态为，由系统过程模型可以预测出基于系统上一状态的现在状态：



其中是基于上一状态的先验估计值，是上一状态的最优解，是当前时刻的控制量。

当得到当前时刻的预测值后，需要计算当前预测值的协方差，即的协方差，这里用表示，其结果如下：



其中是对应的协方差，是系统过程噪声的协方差。

同时我们通过某些仪器设备测量到了当前时刻的值，但测量值依旧存在测量误差，这里认为测量误差符合高斯分布，同样前面式得到的预测值也存在误差，并认为误差符合高斯分布。卡尔曼在预测值和测量值服从正态分布中得到时刻最优化的估计值：



上式中[1]为卡尔曼增益：



得到时刻最优估计值后，还需要计算它的协方差：



自此卡尔曼滤波的五个公式全部给出。下一时刻可以根据和计算，由此进行迭代。

本文中使用MPU6050加速度陀螺仪模块进行姿态平衡，由于模块输出数据存在噪声，且加速度数据的高频噪声严重，而陀螺仪的温漂对角度的估计也存在很大影响，因此这里需要对数据进行滤波，下面结合卡尔曼滤波器进行MPU6050数据的滤波处理。

由于MPU6050会输出当前时刻的X、Y、Z三轴的加速度和角速度值，这里我们仅以其中一轴为例进行滤波器的设计和实现。这里需要说明，陀螺仪输出的角速度因为温漂而变得不准确，而温漂工程上一般认为服从白噪声分布，因此这里不直接估计角速度，而是估计温漂，处理首先给出系统的过程变化公式：





其中是角速度，和是过程噪声，因此式和可写成如下形式：



因此系统过程噪声的协方差为：



这里和的值可以根据实际情况设置，选取不同的和可以设置对陀螺仪的信用度。下面可以计算预测量的协方差：



其中为时刻的最优估计值的协方差。

在计算卡尔曼增益之前给出观测量的表达：



其中和为真实的角度值和温漂值，由于温漂无法直接测量，因此设为0，为测量噪声。接下来给出卡尔曼增益：



其中为测量矩阵，为的协方差矩阵，则当前的最优估计值为：



最后更新协方差：



自此基于卡尔曼的MPU6050滤波器设计完成[3]。

3.2 一阶互补滤波器

这里针对MPU6050的输出数据进行一阶互补滤波器：



其中表征系数，改变该值可以调节滤波性能，为加速度确定的角度值。

3.3 一阶低通滤波器

这里设信号序列为例，给出一阶低通滤波器的公式：



3.4 PID算法

两轮自平衡车的控制包含三个方面：直立控制、速度控制和转向控制。

直立控制采用PD算法，负责小车的平衡，为平衡小车的核心部分，具体公式如下所示：



其中为比例系数，为微分系数，为时刻的相位误差，输出为PWM值。

同理速度控制：



其中为速率误差，和分别为比例系数和积分系数。

最后转向控制如下：



结合以上三个公式：



至此小车的平衡算法介绍完毕[4]。

4、硬件设计

4.1 姿态感知单元

姿态检测模块用于检测角加速度和自平衡车与重力方向的倾角。本文选用的姿态检测芯片为 MPU6050，该芯片是世界上第一款整合了3轴加速度计和3轴陀螺仪的6轴运动跟踪组件。其内部包含数字运动处理器 DMP，时间校准器，拥有丰富的接口如：IIC、SPI，支持主从模式。



图4.1 MPU6050模块实物图

该芯片的主要特性如下：

(1)16 位的 ADC，将陀螺仪和加速度计采集的模拟量以数字量输出；

(2)陀螺仪的测量范围为，加速度计的可测范围为；

(3)1024字节的FIFO缓冲器，能够帮助降低系统功耗；

(4)400KHz的IIC快速模式；

(5)内部嵌入了一个数字输出温度传感器；

(6)输入电压为3.3V；

本文中 MPU6050与STM32的通讯方式为 IIC 通信。IIC 是由数据总线 SDA和时钟总线 SCL 两条串行总线组成。连接到设备上的可以是从机也可以是主机。主机将需要通讯的从机地址发送到地址总线上，从机设备匹配地址，之后对主机发出应答信号。在这里 MPU6050 作为从机使用。

下面给出MPU6050六轴传感器模块的引脚说明：

表4.1 MPU6050模块引脚说明

|  |  |  |
| --- | --- | --- |
| 名称 | 说明 | 与STM32引脚映射 |
| VCC | 3.3V/5V电源说明 | VCC |
| GND | 地线 | GND |
| IIC\_SDA | IIC通信数据线 | PB7 |
| IIC\_SCL | IIC通信时钟线 | PB6 |
| MPU\_INT | 中断输出引脚 |  |
| MPU\_AD0 | IIC从机地址设置引脚：ID：0X68（悬空/接地） ID：0X69（接VCC） |  |

4.2 测距单元

（该模块最后未搭载上小车）

超声波小车测距用于小车预警，防止小车前行撞到物体。本设计使用的是 HC-SR04 超声波模块, 该模块主要包括超声波发射器、接收器、与控制电路。

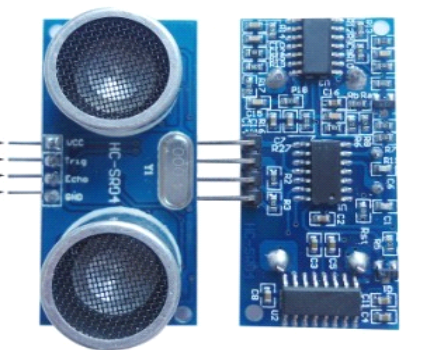


图4.2 超声波测距模块

模块的电气参数如下：

(1)输入电压5V，工作电流为15mA；

(2)测距范围2cm-400cm；

(3)工作频率40KHz；

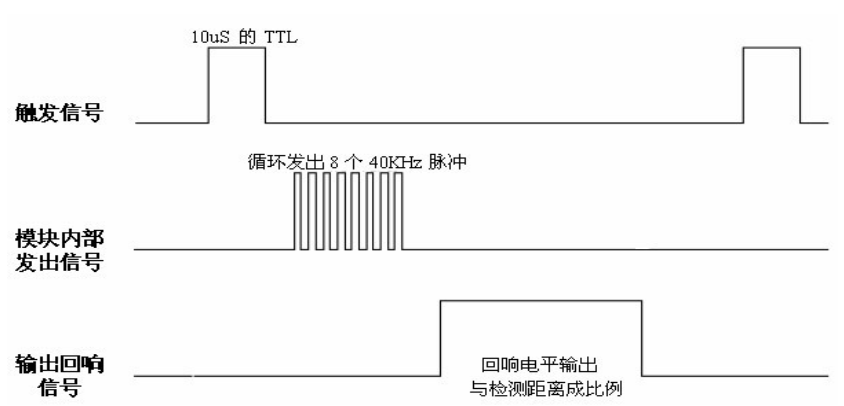


图4.3 超声波测距原理

以上时序图表明你只需要提供一个10uS以上脉冲触发信号，该模块内部将发出8个40kHz周期电平并检测回波。一旦检测到有回波信号则输出回响信号，回响信号的脉冲宽度与所测的距离成正比。由此通过发射信号到收到的回响信号时间间隔可以计算得到距离。

距离=高电平时间\*声速（340M/S）/2；建议测量周期为 60ms以上，以防止发射信号对回响信号的影响。

4.3 无线传输单元

这里选用ATK-BLE01是正点原子推出的一款支持蓝牙4.2协议的低功耗数传模块，无线工作频率为 2.4GHz，模块主从一体，支持多种工作模式、 低功耗模式， 支持数传，支持一对多数据广播，内置 iBeacon 协议，可以作为 iBeacon 设备。

表4.2 模块引脚说明

|  |  |  |
| --- | --- | --- |
| 名称 | 说明 | 与STM32引脚映射 |
| VCC | 3.3-5V电源说明 | VCC |
| GND | 地线 | GND |
| TXD | 串口输出 | PB11 |
| RXD | 串口输入 | PB10 |
| STA | 模块链接状态（连接、断开） |  |
| WKUP | 模块睡眠唤醒引脚  串口升级使能引脚 |  |



图4.4 ATK-BLE01模块

产品特点：

1、 支持 2.0~3.6V 直流电压供电范围；

2、 模块集邮票孔和排针焊接孔， 排针焊接孔方便用户测试；

3、 支持多种工作模式：主设备、从设备、广播者、观察者、 iBeacon；

4、 支持多种睡眠模式，最低功耗 0.5uA；

5、 支持模块发射功率设置，范围： -14dBm 到+8dBm；

6、 支持多种天线形式：板载天线、外置 IPEX 天线座、外置天线；

7、 支持 Android4.3 和 iphone4S 以上版本的手机；

8、 空旷环境传输距离 100 米；

9、 板载状态指示灯，方便观察工作状态；

10、 支持 AT 指令配置参数；

11、 支持串口波特率： 1200-921600， 默认： 115200；

12、 支持外部控制模块睡眠和唤醒；

13、 支持无线唤醒和串口唤醒；

14、 支持上电自动连接、断线自动回连、密码配对连接、用户数据加密；

15、 支持一对多广播、 iBeacon 功能、微信摇一摇周边定位；

16、 支持本地固件升级；

4.4 显示模块

显示模块采用的是 0.96 寸的 OLED 显示屏，其作用是显示小车的运行电压、倾角、电机速度和超声波测得的距离。0.96存的OLED可显示128\*64的点阵，每个点阵都是自己发光，没有背光这一说法，可显示汉字、ASCII、图案等。具体接口信息如下图所示。

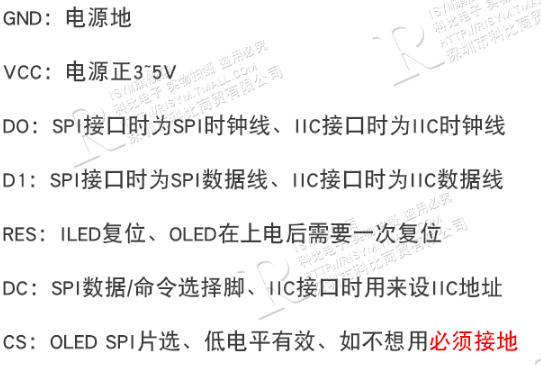


图4.5 OLED接口信息

表4.3 OLED显示引脚连接

|  |  |
| --- | --- |
| 名称 | 与STM32引脚映射 |
| VCC | VCC |
| GND | GND |
| DI | PB13 |
| DO | PB15 |
| DC | PB14 |
| RES | PB12 |
| CS | PB1 |

4.5 驱动模块

驱动电机的驱动芯片这里选择了常用的TB6612FNG驱动芯片。TB6612是双驱动，也就是可以驱动两个电机，驱动电流可达1.2A（峰值3.2A），可以通过IN1和IN2两个引脚控制电机的四种工作状态，电机的速度可以通过多达100KHz的 PWM波进行控制，下面简单介绍下芯片部分引脚的参数。

表4.4 TB6612FNG部分引脚说明

|  |  |  |
| --- | --- | --- |
| 名称 | 说明 | 与STM32引脚映射 |
| VCC | 逻辑电源2.7~5.5VDC | VCC |
| VM | 电机电源不大于15VDC | 电池正极 |
| STBY | 接单片机IO口，清零电机全部停止，置一通过AIN1、AIN2、BIN1、BIN2来控制正反转 | VCC |
| GND | 接电源负极 | GND |
| PWMA | 接单片机的PWM口 | PB6 |
| (AIN1、ain2) | (0,0)停止、(0,1)正转、(1,0)反转 | (PA0、PA1) |
| (AO1、AO2) | 接电机A的两个角 | 电机正负两极 |
| PWMB | 接单片机的PWM口 | PB7 |
| (BIN1、Bin2) | (0,0)停止、(0,1)正转、(1,0)反转 | （PA2、PA3） |
| (BO1、BO2) | 接电机A的两个角 | 电机正负两极 |

4.6 电机模块

这里我们选择带编码器N20直流减速电机，具体的参数如下如所示。



图4.6 带编码器的直流减速电机

这里介绍编码器关于电机转向的判断，本项目中并未使用定时器捕获电机的转速，而是采用简单外部引脚上升沿中断触发，计算上一次与本次触发的时间差来表示电机运转的周期T。由图4.6可知，编码器配置了两个霍尔传感器，且两个传感器之间的距离小于黑色磁片圆周长的一半，这里定义从磁片侧看去的左右两个霍尔传感器为S1和S2，考虑电机正转和反转两种情况：

正传（顺时针）：当磁片触发S1时，在S1处计算上次与当前的时间差，得到电机的运转周期T1=T，同时刻在S1处我们也可以得到霍尔传感器S2的当前时间与S2上次触发的时间差T2，可知T2时长小于电机的运转周期T，即T2<T，但满足T2>T1/2即T2>T/2。

反转（逆时针）：同样磁片触发S1时，T2<T1/2即T2<T/2。

因此可以根据T2与T1/2的相对大小进行正反转的判断。

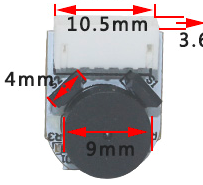


图 4.7 霍尔传感器的位置

4.7 核心系统

这里选用STM32F103C8T6作为核心控制单元，它内核是32位ARM的Cortex-M3，主频为72MHz，拥有2\*SPI、2\*IIC、3\*USART、USB和CAN接口，32个IO接口，2个12bit的ADC，可以满足项目实现。

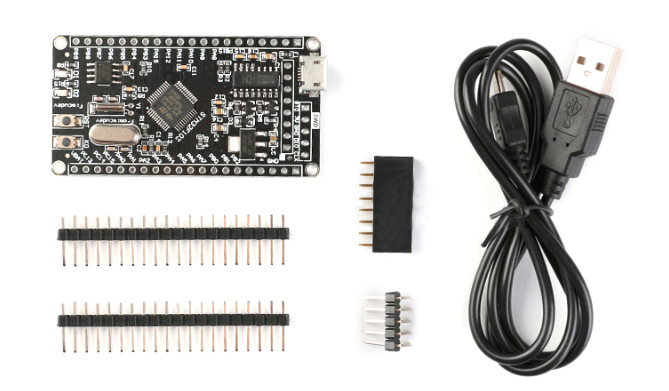


图4.8 主控芯片

5、软件设计

下面给出总体的软件架构。



图5.1 软件架构

5.1 通信模块

项目使用蓝牙作为通信模块，该模块主要有两个功能：1、数据传输；2、控制功能。蓝牙设置主从模式，PC端为主，小车端为从，由PC发送控制命令，小车端反馈相应的数据上来。

表5.1 通信模块功能

|  |  |
| --- | --- |
| 功能 | 映射 |
| 数据传输 | [三维角速度、三维加速度、三维角度、速度] |
| 控制功能 | 设置正向速度(HEX)：[CE,01, 一个字节的速度] |
| 设置反向速度(HEX)：[CE,02, 一个字节的速度] |
| 设置右转方向(HEX)：[CF,01, 一个字节的方向增量] |
| 设置左转方向(HEX)：[CF,02, 一个字节的方向增量] |

数据传输主要功能是将MPU6050三维加速度和三维角速度的数据上传到PC端，通过Matlab进行滤波算法的仿真和调试，选择合适的滤波算法进行姿态融合，从而避免频繁地在代码中更改参数、烧录和调试，节省时间。这里的数据传输使用的是STM32F103开发板中的MPU6050六轴传感器时延例程中的usart1\_report\_imu函数，文件参考资料中附上了Matlab的相关仿真程序MPU6050\_Signal\_Process.m，仿真数据是通过串口调试助手得到的，具体的仿真效果如下图所示。



图5.2 融合算法仿真图

5.2 自平衡模块

自平衡模块需要实现小车的平衡功能，即角度PID环、速度PID环、转向PID环三者的结合。

表5.2 平衡车功能函数

|  |  |
| --- | --- |
| 输入 | 三维角度、电机速度、转向角度 |
| 输出 | PWM波 |

下面给出自平衡小车核心算法流程：



图 5.2 平衡小车的核心算法流程

5.3 显示模块

显示模块主要实现的功能：1、显示三维角度；2、显示当前的速度；3、显示电量。这里不做过多介绍。

5.4 电脑遥控

此项目未使用手机端进行遥控，选用了基于Python的电脑键盘遥控的方式。具体实现的方式为：电脑通过USB转串口设备连接主蓝牙，该蓝牙继而连接自平衡小车上从蓝牙，因此可以在电脑上控制小。具体的实现代码已在附件参考资料中，注意电脑需要安装相应的USB转串口的驱动才可使用。

6、总结

6.1 初步执行计划

为了更好的完成该项目，结合之前的项目经验，下面给出初步的执行计划日程表。

|  |  |  |
| --- | --- | --- |
| 执行内容 | 时间规划 | 是否完成 |
| 前期调研 | 2020.11.03-2020.11.06 | √ |
| 方案撰写 | 2020.11.07-2020.11.09 | √ |
| 硬件平台和实验平台搭建 | 2020.11.10-2020.11.14 | √ |
| 算法实现和调试 | 2020.11.15-2020.11.17 | × |
| 算法实现和调试 | 2020.11.15-2020.12.05 | √ |
| 完成总结 | 2020.12.05-2020.12.06 | /(ㄒoㄒ)/~~ |

6.2 经验总结

成品：

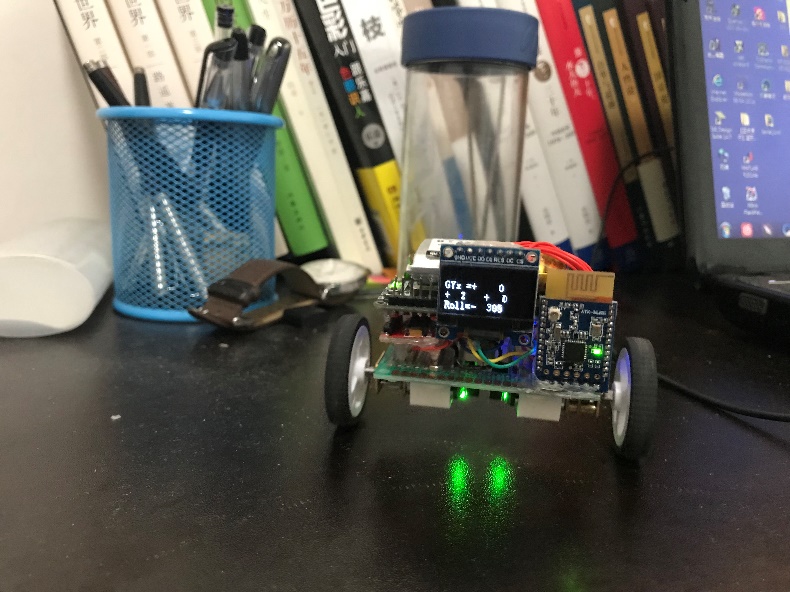


图6.1 实物图

这里主要总结下PID调试的经验。

1. 关于电机选型是否合适的判断：电机选型主要考虑电机的转速，一般对于小型的平衡车转速大于300rpm就合适了。在调试中，若小车在倾斜30°的情况下，依旧可以依靠电机全速运转将小车扳回至相反的方向，则可证所选电机可以满足自平衡的功能。
2. 关于PID角度环的调试问题：
   1. MPU6050输出的角度：由于MPU6050输出数据包含加速度和角速度，而加速度虽然反应快但噪声大，角速度也存在零漂等问题，因此需要对角度进行滤波，可选择的有一阶互补滤波、卡尔曼滤波等，本项目最终选择了一阶互补滤波。由于滤波会导致输入数据与滤波输出数据存在一定的时延，而这时延若是过大，导致滤波后的数据变化跟不上小车的实际角度的变化，会造成小车自平衡难以实现。一般而言，滤波造成的时延不被肉眼察觉即可（可人为改变小车的倾斜的角度，看轮子的转向变化是否有明显延时）。
   2. 角度PID调试中，第一需要选好小车的平衡点，即平衡角度，这个至关重要；第二调好P和D后发现小车不能静止，且向某一方向缓慢加速前进直至倒地，可以考虑加上I；第三角度PID调试好之后，小车可以依旧不能静止，朝某一方向前进，但不会倒下，这时候可以考虑加上速度环，加上速度环后依旧不能静止，可考虑改变下最开始设置的平衡角度。
   3. 遇到反常的情况，多考虑自己的问题，是否代码逻辑有问题，是否底层配置有问题，可以排除干扰、定位问题。

7、参考文献

[1]<https://blog.csdn.net/victor_zy/article/details/82862904?utm_medium=distribute.pc_aggpage_search_result.none-task-blog-2~all~first_rank_v2~rank_v28-3-82862904.nonecase&utm_term=%E5%8D%A1%E5%B0%94%E6%9B%BC%E6%BB%A4%E6%B3%A2%E7%9A%84%E8%A7%82%E6%B5%8B%E7%9F%A9%E9%98%B5&spm=1000.2123.3001.4430>.

[2] <https://www.jianshu.com/p/d3b1c3d307e0>.

[3]基于STM32的两轮自平衡车设计和实现.

[4]两轮自平衡小车控制算法的研究和优化.