基于蓝牙的智能跟随小车

周天杨　　陈明明

第一部分　设计概述

1.1 设计目的

本次项目是基于炬芯平台设计的一款蓝牙智能跟随小车。随着网络速度的提高，智能 IOT 设备蕴藏着巨大的市场需求。我们根据日常生活中行李自动给随，和超市购物车自动跟随等场景，设计一款智能跟随设备。目前开发的产品都是基于 UWB 技术的，成本较高，对应用场景的推广有很大的局限性。基于蓝牙 BLE 开发此项目，可以将成本降低到很低，且能结合手机使用，具有很广阔的应用前景。

1.2 应用领域

主要运用在工业和消费领域。工厂工人使用推车运送货物时，能够解放双手，更好的观察周围的环境，减轻工作压力。当出差时，大包小包的行李箱为旅行体验减分时，自动跟随行李箱可以让旅客更加关注周围的风景，行李箱会自动跟随，在超市购物时，也不用一直推着购物车，可以提升购物体验。此项目对人和物的简单交互控制，有巨大的市场空间。

1.3 主要技术特点

蓝牙 BLE 处于连接态，项目分为简单版和复杂版。简单版通过蓝牙串口，手机端发送控制小车上下左右的命令，开发板接收到后，调节相应的 PWM 波输出，控制小车移动。复杂版，通过手机定时发送手机内方向传感器的数据，结合板载 IIC 协议的方向传感器进行数据比对，当手机端发生方向偏移时，板载方向传感器会将信息传至 CPU，进而控制 PWM 波占空比进行差速转向。

蓝牙包中的 RSSI 值负责控制距离，当距离低于某个 RSSI 值时前进，高于某个 RSSI 值后退，处在某个 RSSI 范围区间内时，小车静止。实现自动跟随。

1.4 关键性能指标

（1） 根据手机蓝牙串口的收发信息，进行前进，后退，左转和右转。
（2） 小车能根据方向传感器的比对，进行自动转弯操作。
（3） 小车能根据蓝牙 RSSI 值的范围，进行距离的大概测算，控制小车前进，终止，后退

1.5 主要创新点

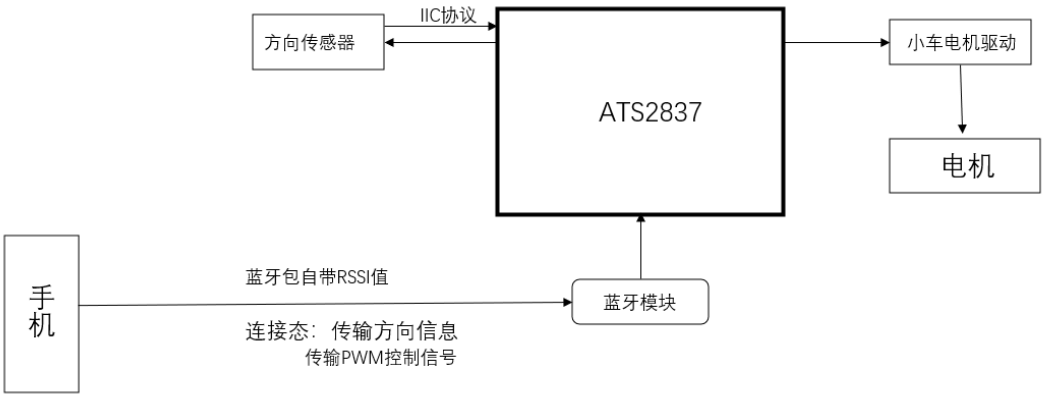
（1） 提出用蓝牙 BLE 模块来实现小车自动跟随，具有轻量级，低成本的优点。
（2） 提出用方向传感器比较的方式控制方向，方向可控制的十分精准，也充

分利用了移动设备，如手机的优点。

（3） 提出用蓝牙 RSSI 测距，由于板载只有单天线，所以使用此设计方法。

第二部分 系统组成及功能说明

2.1 整体介绍



手机在 BLE 连接态发送控制信息，和方向传感器的角度信息，通过蓝牙串口传输给 CPU。CPU 根据控制信息，直接驱动 PWM 波控制电机运行。CPU 比对手机的角度，修正方向，并驱动 PWM 波进行左右方向转换。当手机方向传感器的信息和小车方向传感器的角度相差±3°时，停止差速转动。小车方向传感器遵从 IIC 协议。CPU 解析接收数据包 RSSI 值，通过公式运算控制好距离，在少于-70dB 时前进，大于-40dB 时后退，介于两者范围之内小车停止运动。

2.2 各模块介绍

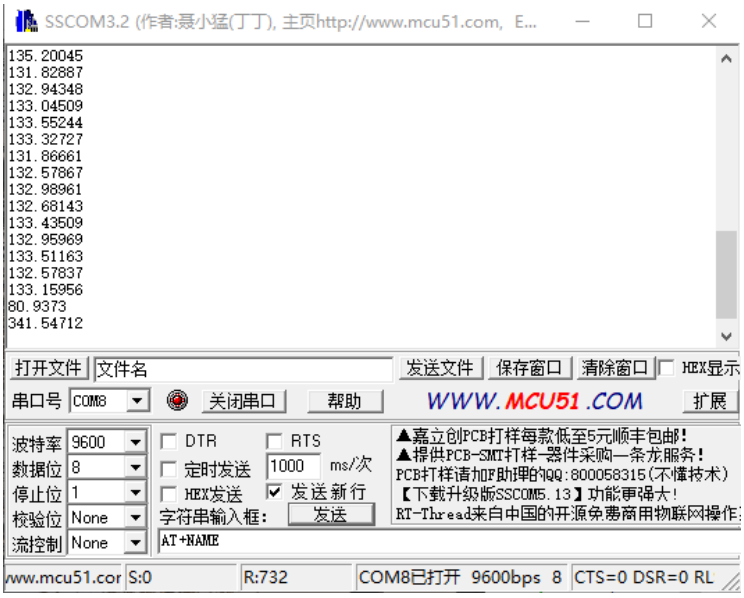
手机主要时设计一个 app,可以连接 BLE 设备。通过 BLE 收发串口协议，根据特定设备的 UUID 号，进行读写服务。

CPU 主要是处理数据，并控制引脚的 PWM 波驱动小车前进，后退，转弯。方向传感器和手机发送的角度信息进行比对。

电机驱动输出电流，控制小车运动。

第三部分 完成情况及性能参数

App 开发工作已经完成。



通过串口调试，可以发现角度信息能够比较精确的发送。
小车的调试，见发送的视频。
目前通过手机控制小车移动已经可以实现，距离自动控制还做的不够好。

第四部分　总结

4.1 可扩展之处

1.RSSI 算法还要再精进，实际 RSSI 方法测距的稳定性不行。需要移植一种算法使得距离再精确一些。

2.实际手机可把连接态改成广播态，通过广播发送信息，小车的蓝牙改成扫描态，这样就可以同时控制多台设备的自动跟随。

4.2 心得体会

通过此次比赛，熟悉了交叉编译的流程。实现了对平头哥 ATS2837 这款芯片的运用，但实际操作过程中，内部驱动有些问题，调试花了大功夫，以至于时间紧张，并未完成所有目标任务。这点还需要注意。

第五部分 参考文献

[1] 肖晓兰，黄海峰等，一种自动跟随手机行走的小车.[D].广东：广东工业大学
2016.4

［2］ 谭玉林．轮式机器人的分析与设计［D］．西安：西华大学，2010：１０－１１．

［3］ 姚文详，宋岩．ＡＲＭ Ｃｏｒｔｅｘ－Ｍ３权威指南［Ｍ］．北京：北京航空航天大学出版社，2009．

［4］ 董景新，赵长德．控制工程基础［Ｍ］．北京：清华大学出版社，2009．

［5］ 任志斌．电动机的 ＤＳＰ控制技术与实践［Ｍ］．北京：中国电力出版社，2012．

［6］ 曲永印，白晶．电力电子技术［Ｍ］．北京：机械工业出 版社，2013．

［7］ 巫付专，王晓雷．控制电机及其应用［Ｍ］．北京：电子工业出版社，2008．

［8］ 武永亮．Ａｎｄｒｏｉｄ开发范例实战宝典［Ｍ］．北京：清华大学出版社，2013．

第六部分 附录

重要代码、推导过程等不便于在正文中体现的内容

6.1PWM 驱动

由于官方提供的 PWM 驱动的 API 存在一定的问题，因此本基础驱动模块为平头哥指导老师自行编写后提供。如图 6.1 所示，为提供调用的 PWM 波设置函数。其原理为设置其内部寄存器，如图 6.2 所示。

```
void verimake_pwm_set(u32_t pwm_channel,bool Is_enable,bool Is_high_or_low_active,bool
Is_breath_or_normal,u32_t pwm_T_times,u32_t pwm_duty)
{
switch(pwm_channel)
{
    case PWM_CHANNEL_0:
    verimake_pwm_0_set
(Is_enable,Is_high_or_low_active,Is_breath_or_normal,pwm_T_times,pwm_duty);break;
    case PWM_CHANNEL_1:
    verimake_pwm_1_set
(Is_enable,Is_high_or_low_active,Is_breath_or_normal,pwm_T_times,pwm_duty);break;
    case PWM_CHANNEL_2:
    verimake_pwm_2_set
(Is_enable,Is_high_or_low_active,Is_breath_or_normal,pwm_T_times,pwm_duty);break;
    case PWM_CHANNEL_3:
    verimake_pwm_3_set
(Is_enable,Is_high_or_low_active,Is_breath_or_normal,pwm_T_times,pwm_duty);break;
    case PWM_CHANNEL_4:
    verimake_pwm_4_set
(Is_enable,Is_high_or_low_active,Is_breath_or_normal,pwm_T_times,pwm_duty);break;
    case PWM_CHANNEL_5:
    verimake_pwm_5_set
(Is_enable,Is_high_or_low_active,Is_breath_or_normal,pwm_T_times,pwm_duty);break;
    case PWM_CHANNEL_6:
    verimake_pwm_6_set
(Is_enable,Is_high_or_low_active,Is_breath_or_normal,pwm_T_times,pwm_duty);break;
    case PWM_CHANNEL_7:
    verimake_pwm_7_set
(Is_enable,Is_high_or_low_active,Is_breath_or_normal,pwm_T_times,pwm_duty);break;
    case PWM_CHANNEL_8:
```

图 6.1

```
void verimake_pwm_0_set(bool Is_enable,bool Is_high_or_low_active,bool Is_breath_or_normal,u32_t
pwm_T_times,u32_t pwm_duty)
{
    PWM0_CTL=0;
    PWM0_CTL|=Is_enable<<28;
    PWM0_CTL|=Is_high_or_low_active<<27;
    PWM0_CTL|=Is_breath_or_normal<<26;
    V_CMU_PWM0CLK=pwm_T_times;
    PWM0_CTL|=pwm_duty;
}
```

图 6.2

6.2 蓝牙透传

当蓝牙模块接收到数据后，可以通过读取该寄存器的方式，获得该数据。如图 6.3 所示。ble_buf_not_decode 指向数据地址，*ble_buf_not_decode 为 char 类型的接收到的数据。

```
static ssize_t write_wp_tx(struct bt_conn *conn, const struct bt_gatt_attr *attr,
                           const void *buf, u16_t len, u16_t offset, u8_t flags)
{
#if 1
        SYS_LOG_INF("data len %u\n", len);
    ble_led(buf, 1, len, 16, 0);
        print_buffer(buf, 1, len, 16, 0);
        ble_buf_not_decode = (char*)buf;
        ble_len = len;
        ble_data=*ble_buf_not_decode;
        data=(int)ble_data;
//    sysble_send(&ble_data, len, true);
//    sysble_send(&ble_data, len, false);
        return len;
```

图 6.3

6.3 系统设计

根据接收到的蓝牙数据，可以控制小车的方向以及移动速度。如图 6.4 所示。

```
void verimake_loop()
{
        int speed=0;
    PWM_init();

    while(1)
    {
        os_sleep(OS_MSEC(1));//wait
        switch(*ble_buf_not_decode)
        {
        case 0x01:
                {straight(speed);break;}
        case 0x02:
                {forward(speed);break;}
        case 0x03:
                {left(speed);break;}
        case 0x04:
                {right(speed);break;}
        case 0x05:
                {
                speeed=speed+5;
                if(speed<=255)
                        speed=255;
                break;
                }
        case 0x06:
                {
                speeed=speed-5;
                if(speed<=0)
                        speed=0;
                break;
                }
        case 0x07:
                {
                speed=127;break;
                }
        default:
                {
                right_stop();left_stop();break;
                }
        }
    }
}
```

图 6.4