浙江大学

探究性实验项目总结报告

课程名称　　智能控制系统设计与实践

实验项目　　基于摄像头和电磁传感器的智能循迹控制系统设计

姓　　名　　贾茗凯

学　　号　　3170105806

专　　业　　自动化(机械电子工程交叉创新平台)

时　　间　　2020 年 08 月 30 日

一、 实验目的和要求

1. 了解车模构成，掌握车模组装方法；
2. 掌握 MCU 的驱动和软件开发；
3. 掌握常用传感器的驱动方式，如摄像头、编码器、陀螺仪等；
4. 掌握 PCB 的绘制和电路板的焊接、调试；
5. 掌握图像校正、提取黑线、赛道元素识别的方法；
6. 掌握模糊控制、PID 控制的原理和参数调试方法。

二、 实验内容

2.1 控制器选型

方案一：采用 LPC54606 芯片。该芯片采用 Cortex-M4 内核，主频 180MHz，SRAM 为 200K，Flash 为 512K，对于小车图像处理算法性能不足。

方案二：采用 i.MX RT1064 芯片。该芯片是 i.MX RT 系列，由 NXP 半导体公司推出，是一款跨界处理器芯片。该系列下又包括 i.MX RT1020、i.MX RT1050 及 i.MX RT1060 等子系列芯片。该芯片采用了微控制器的内核 Cortex-M7，对中断响应更快，具有良好的实时性，从而具有应用处理器的高性能及丰富的功能，又具备传统微控制器的易用、实时及低功耗的特性。其主频高达 600M，且板载 4M 大容量片上 Flash 和 512KB 的 RAM，满足程序开发的需求。

因此最终选择方案二，采用逐飞 RT1064 核心板。

2.2 摄像头选型

方案一：采用 OV7725 二值化摄像头。虽然其数据量小，易于单片机处理，但由于竞赛中摄像头的高度要求 10cm，该摄像头已经不再满足新的赛道条件的需求。

方案二：采用 MT9V032 灰度摄像头。其拥有 140°广角视域，帧率高。因此图像更新速度快，采集信息量大，满足比赛需求。

因此结合上述分析，最终选择方案二，摄像头采用总钻风 MT9V032，帧率 150FPS。

在实际应用时，摄像头帧率设置过高会导致图像变暗，且过高的帧率会超出程序控制周期，即图像更新的信息在控制时可能不被采用。此时可通过配置摄像头增益等参数对图像进行优化，根据实验效果选择较优的配置。

2.3 屏幕选型

方案一： 采用 OLED 显示屏。传输数据量小，实时性高。但分辨率较低，尽可显示二值化图像。

方案二： 采用 IPS 显示屏。分辨率更高，能显示摄像头原图数据，但是数据是 16 位灰度数据，传输数据量非常大。在显示图像和变量时，由于软件 SPI 延时，图像处理程序仅为 20FPS，不满足智能车高速运行时的帧率需求，所以 IPS 屏仅能在静止调试时使用。

因此最终选择方案二，采用 IPS 显示屏。

2.4 智能车系统整体方案

1. 采用 MT9V032 摄像头，以 150fps 的帧率拍摄赛道图像。图像数据通过 DMA 自动传输（场中断传输和行中断传输）给 RT1064，然后运行程序对图像进行反光点去除、边缘检测+二值化预处理、校正、分割并提取黑色边线，并经过路径规划得到小车的预期路径，并判断赛道类型。
2. 由规划好的路径，选择图像合适位置的参考行，计算出偏差，用模糊 PD 控制舵机转向；当进入坡道时，由于拍摄到赛道外图像，故切出摄像头状态，使小车保持直行，离开坡道恢复。
3. 通过编码器检测车速，RT1064 输入捕捉功能进行脉冲计数，2ms 的 pit 中断测量。通过脉冲计数及时间，可计算得到小车的实时速度。通过路径偏差计算出预期速度，并利用 PI 控制以及 bang-bang 控制算法调节电机输出，从而实现小车速度控制。
4. 采用陀螺仪进行坡道辅助判断。记小车车头朝向为 x 轴正方向，通过 I2C 协议向陀螺仪 ICM20602 的相关寄存器中实时读取当前 y 方向的角速度值，以获得小车的翻转角速度，从而判断当前是否正在经过坡道或者颠簸。由于小车撞到路肩时同样可能出现翻转角，因此加入防误判算法。
5. 向 UART 转 Wi-Fi 模块以 150000 的波特率发送串口数据。模块将数据传输回上位机，实现实时监控，从而获得小车运行过程中的各项观察指标，便于后续参数调节。
6. 采用拨码开关、按键以及 IPS 显示屏辅助进行控制策略选择和参数设置。

系统框图如下：

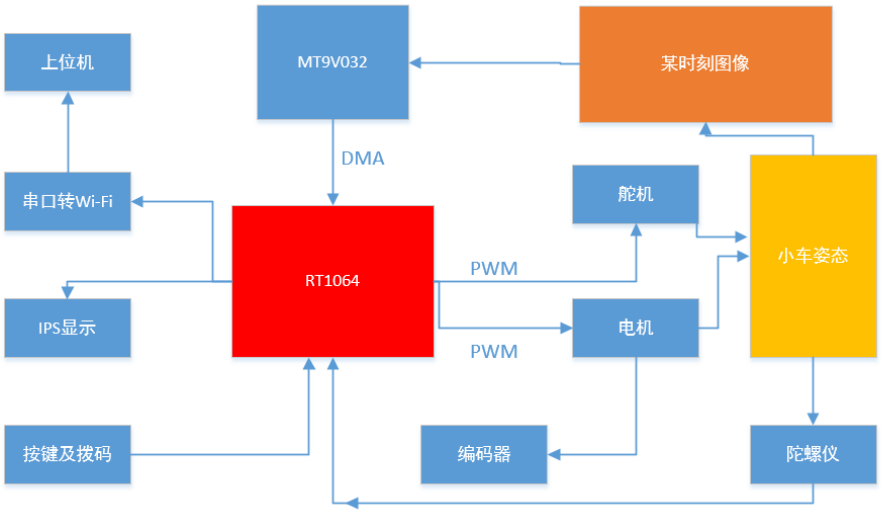

图 1 系统框图

三、 技术要点

第一节 机械设计

由于机械决定了小车的最高速度，因此机械设计较为重要。机械的变化可以提升小车的速度上限，机械的稳定性非常重要，以下为具体的机械设计过程。

1.1 重心

小车的重心是整个机械最重要的两个因素之一。

- 重心要尽可能的低，但不能擦到地面。在高速过弯的时候，重心较高的车更有可能会抬轮，即为弯侧的两个轮子由于离心力作用抬了起来。此时只有两个轮子着地，只有很小一部分接触了地面，因此容易发生侧滑、甩尾或者跳轮（轮子和地面不停撞击）等现象。
- 调整重心的前后位置。重心太靠前，后轮的抓地力不够，会发生甩尾；重心太靠后，前轮抓地力不够，会发生侧划（前轮滑动）的现象。电池对小车的重心影响最大，因此电池的位置很关键。
- 整车的重量要尽可能的轻，电机的加减速性能会比较好。前轮侧滑可以加几个配重块以改善。

1.2 舵机和前轮

前轮转向部分主要由舵机、舵机摆杆、拉杆、主销、主销座和轮毂构成，是小车中结构最为复杂的机械。

- **舵机：**摆杆不宜太短，一般取的尽量长，以缩短舵机滞后时间。拉杆一般装成略带 V 的形状，在大弯的时候获得最大的拉力。
- **主销：**调节拉杆的长度，以调节主销的前束角；拧主销座上的两个螺丝，以调节主销的内外倾角。当前轮左后晃动幅度较小时，倾角要小一点，当前轮由于损耗左右晃动幅度很大时，倾角可以稍大一点。因此在弯中行驶时总有一个前轮是整个轮子都贴在地面上的，从而增大了前轮的摩擦力。

1.3 摄像头安装

- **摄像头架的安装：**靠后放可以将视野范围扩大。小车在大弯中靠外圈行驶时，很可能会造成摄像头看不到或只看到一点点边线，而摄像头往后放可以让摄像头视野扩大。
- **摄像头的高度、角度：**摄像头的高度角度，主要决定了摄像头能看到的最远端和最近端。一般来讲，摄像头的最近端越靠近车头越好，最远端是越远越好。此外前瞻行（一般处在车前 40cm~70cm 左右）尽量的靠近图像的下半边，因为图像的下半边最稳定，不容易发生边界跳动的情况。
- **摄像头支架：**在小车行驶中，摄像头的杆子会晃动。为了减小这个晃动，使用两根细的碳素杆把主杆给支起，因此主杆强度不需要过大。

第二节 速度控制

2.1 速度的获取

采用编码器加装在后轮差速器齿轮上进行测速。编码器分为三线编码器和四线编码器。三线编码器只能测量速度，并不能测量速度的方向。而四线编码器（如 512 线 ABI 编码器）可以测量速度的方向。

四线编码器由电源线、地线和 A，B 两相的输入组成。A，B 两相处于正交编码状态，即正转或者反转时 A，B 相的相位差会不同。多数单片机具有正交编码功能，可以自动对 A，B 相进行解码（计数），计数结果为有符号量。

计数值可通过运算转换成距离，使用 PIT 定时中断可获取时间，相除可得速度值。程序初始化完成后经过一定的时间（2ms），进入一次中断，在中断中读取计数值，并转换为速度值。

系统中除了 PIT 中断，还存在着很多中断，其中，摄像头的场中断和 DMA 中断是系统中最重要的，所以优先级较高；PIT 中断选择一个比较靠前的合适位置即可，虽然可能被打断，但测速的误差可以接受。

2.2 速度控制算法

在速度较低时，速度控制的重要性不显，但是当小车速度越来越高时，速度控制显得尤为重要。并且当速度的性能要求较高时，传统 PI 控制算法已经满足不了需要。故我们对速度控制策略进行了优化。

根据经验，在速度 2m/s 以下，基本不需要进行电机速度控制，在速度 2m/s

以上，则必须加入速度控制算法[4]，例如增量 PI 控制、PI 差速控制、bang-bang 控制、增量 bang-bang 差速控制、定 PWM 差速控制等。

首先是速度控制周期的选取，一般图像采集一场的周期是 6~7ms，速度控制周期应与之相近。速度控制周期太长，会导致控制不及时，速度会产生很大的超调；周期太短，由于在一个周期内各个量变化较小，可能会导致速度测量的误差影响增大。

速度控制一般采用 PI 算法，并且是一个严格意义上的随动系统。首先根据路径计算出当前的期望速度。我们采用与舵机偏差相似的计算方法。由于电机作用的滞后比舵机更加严重，其应该具有更远的前瞻性。获得期望速度后，与测得的速度比较，得到偏差 error，然后就用标准的 PI 控制对电机进行控制。

P 分量：为使小车获得更高的加速度，一般需要将 P 调大，但会导致减速时极易产生超调。如小车刚入弯时，速度明显减慢，但小车在弯心时，由于电机控制超调，小车反而出现了速度回升。所以 P 要满足在可承受的超调下尽量大。

I 分量：I 分量的目的是消除静态误差。为了防止积分饱和，需要给 I 分量加上限幅。但是加入 I 分量后，也存在着一定的静差。

补偿：由于电机存在着反电动势，当转速越快时，反电动势就越大，在整体表现上就显得不灵敏，该反电动势不利于电机控制。为补偿该电动势，可在电机的输出中加一项正比于速度的项来补偿反电动势，再加一项常数项来抵消其他阻力。由于这些补偿属于正反馈，所以需要进行适度调节。

2.3 差速控制算法

由于车模采用双电机驱动，因此采用均速去设定电机的目标速度会导致和经典的四轮车转向模型存在矛盾，导致舵机转向的不灵敏，相当于舵机系统滞后性更强。

为了实现真车类似的差速结构，参考阿克曼转向几何原理，汽车在转弯时，内、外侧后轮行驶距离不同，而两者的行驶时间却相同，因此两者时间存在差速问题，如果采用均速去设定电机的目标速度，会导致和经典的四轮车转向模型存在矛盾，导致舵机转向的不灵敏，相当于舵机系统滞后性更强。所以根据舵机的打角来对过弯时后轮两电机的速度进行差速控制是必要的。

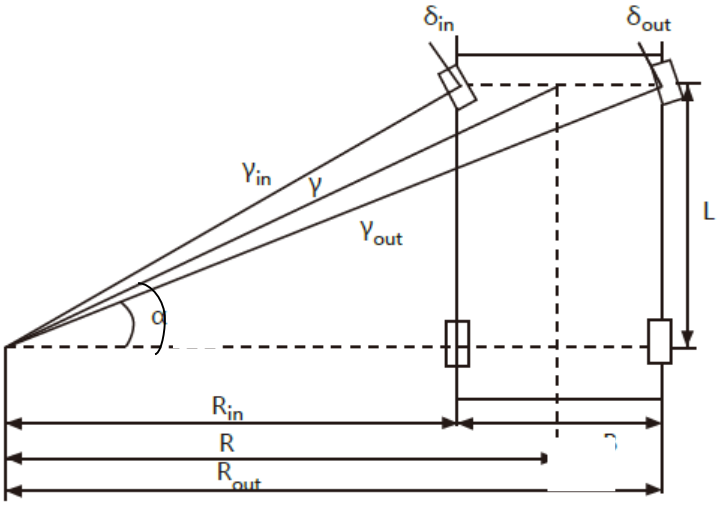

图 2 阿克曼转向模型

根据阿克曼转向模型，如图 7 所示，在车轮与地面不发生打滑的情况下，设小车后轮中心点运动速度为 v，则小车行进角速度为

$$\omega = \frac{v}{R} \qquad \text{(公式 1)}$$

由于小车为刚体结构，小车上各点运行角速度相同，故有小车左右轮速v_1，v_2分别为

$$v_1 = \omega R_{in} \tag{公式 2}$$

$$v_2 = \omega R_{out} \tag{公式 3}$$

由阿克曼转向模型几何关系可知

$$R = \frac{L}{\tan\theta} \tag{公式 4}$$

于是有两轮差速

$$v_2 - v_1 = v\frac{W}{L}\tan\theta \tag{公式 5}$$

也即

$$\begin{cases} v_1 = v\left(1 - \dfrac{W}{2L}\tan\theta\right) \\ v_2 = v\left(1 + \dfrac{W}{2L}\tan\theta\right) \end{cases} \tag{公式 6}$$

在实际差速控制中，还需要考虑到小车在弯道时的设定速度与实际速度的不匹配，舵机打角的滞后性，舵机饱和，电机 PI 控制的滞后性以及电机 bang-bang 控制的稳定性。故对差速串以分段线性的 P 来控制，即

$$\begin{cases} v_1 = v\left(1 - K_p \dfrac{W}{2L}\tan\theta\right) \\ v_2 = v\left(1 + K_p \dfrac{W}{2L}\tan\theta\right) \end{cases} \quad \text{(公式 7)}$$

第三节 其他

此外，还有电路设计、图像信息获取、方向控制、电磁信息获取、其他传感器等其他技术要点，此处不再赘述。

四、 实验成果

- 此次比赛中，获得全国大学生智能汽车竞赛（浙江赛区）第七名的成绩。
- 此次比赛中，共历时 11 个月，起止时间：2019.09.20-2020.08.10
- 此次比赛中，共撰写 RT1064 嵌入式 C 语言代码 2w 行（修改的库函数除外）。最终输出 C 语言代码 1.1w 行（库函数除外）。
- 此次比赛中，共撰写 Matlab 仿真代码 1k 行，Simulink 仿真工程 2 份。
- 此次比赛中，共输出比赛日志 100 天共 1.2k 行 2w 字。
- 此次比赛中，共输出电路板原理图及 PCB 工程文件 8 组。共输出电路板 100 余块，成品电路 20 余块。
- 此次比赛中，共输出 1w 字技术报告 1 份。

五、 实验体会

- 我本次主要参与整体搭建，同时实现在视觉识别的基础上，实现运动控制。通过本次实验，我更加深入的学习了工程知识，在运动学、动力学、机器视觉的应用上加深了认识。运动控制中，我对车辆控制的相关算法进一步熟悉，完成了 PID 以及 bang-bang 控制的代码编写，并将舵机控制前瞻与速度耦合起来，实现了较好的控制效果。提升了自己对于智能控制系统的设计与实践能力。在机械设计方面，对车辆各个机械固件的组装、紧固以及机械结构设计工作也进一步加强了我们的设计与实践能力。

- 在实践的过程中，虽然我们各自负责不同的模块，但车辆的整体调试是一个系统工作，各个控制器、常量参数都会对小车动态产生影响，本次实验深度加强了我们对经典控制理论、对模糊控制等控制方法的实践认识，同时在此基础上对嵌入式软件设计又有了新的理解。

参考文献

[1]卢守义,万星,卜令坤.基于 CMOS 摄像头的智能车设计[J].自动化应用,2018(04):138-140.

[2]李永,冯伟峰,李思光,王俊人.基于 MT9V032 摄像头的智能车软件设计[J].花炮科技与市场,2020(01):250.

[3]陈国定,张晓峰,柳正扬.电磁智能车电感排布方案[J].浙江工业大学学报,2016,44(02):124-128.

[4] Sui Jinxue. " NXP Cup Intelligent car Design and Example Tutorial [M]. Beijing: Electronics Industry Press. 2018.8.

[5] C. Kinnaird, "Digital adjustment of DC motor drive circuit parameters," 2016 IEEE Dallas Circuits and Systems Conference, Arlington, TX, pp. 1-4, 2016.

附 A　模型车的主要技术参数说明

改造后的车模总重约 882 克，长约 25 厘米，宽约 19.5 厘米，高约 11.5 厘米（其中摄像头中心高度约 9.5 厘米）。

车模静止状态下电路平均功耗 1.75W，电容总容量 2mF。

车模包含四类传感器：摄像头（1 个）、编码器（2 个）、陀螺仪（1 个）以及电磁传感器（含 4 个电感）。

除车模自带的驱动电机以及舵机外，未使用其它的伺服电机。

赛道信息检测频率与摄像头采样频率相同，约 150Hz；检测精度良好。

附 B　主要代码

1 大津法图像二值化阈值计算

```
uint8_t GetOSTU(void)
{
    int16_t i, j;
    uint32_t Amount = 0;
    uint32_t PixelBack = 0;
    uint32_t PixelIntegralBack = 0;
    uint32_t PixelIntegral = 0;
    int32_t PixelIntegralFore = 0;
    int32_t PixelFore = 0;
    double OmegaBack, OmegaFore, MicroBack, MicroFore, SigmaB, Sigma; // 类间方差;
    int16_t MinValue, MaxValue;
    uint8_t Threshold = 0;
    uint16 HistoGram[256]; //
    int16_t Mincount = 0, Maxcount = 0;

    for (j = 0; j < 256; j++)
    {
        HistoGram[j] = 0; //初始化灰度直方图
    }

    for (j = START_LINE; j < CAMERA_H; j++)
    {
        for (i = 0; i < CAMERA_W; i++)
        {
            HistoGram[Image_Use[j][i]]++; //统计灰度级中每个像素在整幅图像中的个数
        }
    }
```

```
28.
29.     //获取最小灰度的值
30.     for (MinValue = 0; MinValue < 256 && HistoGram[MinValue] <= 5; MinValue++)
31.     {
32.         Mincount += HistoGram[MinValue];
33.         HistoGram[MinValue] = 0;
34.     }
35.     //获取最大灰度的值
36.     for (MaxValue = 255; MaxValue > MinValue && HistoGram[MaxValue] <= 15; MaxValue--)
37.     {
38.         Maxcount += HistoGram[MaxValue];
39.         HistoGram[MaxValue] = 0;
40.     }
41.     //滤除反光点
42.     for (j = START_LINE; j < CAMERA_H; j++)
43.     {
44.         for (i = 1; i < CAMERA_W; i++)
45.         {
46.             if (Image_Use[j][i] > MaxValue - 8)
47.             {
48.                 Image_Use[j][i] = MaxValue - 8;
49.             }
50.         }
51.     }
52.
53.     HistoGram[MaxValue] += Maxcount;
54.     HistoGram[MinValue] += Mincount;
55.     if (MaxValue == MinValue)
56.     {
57.         return MaxValue; // 图像中只有一个颜色
58.     }
59.
60.     if (MinValue + 1 == MaxValue)
```

```
61.     {
62.         return MinValue; // 图像中只有二个颜色
63.     }
64.
65.     for (j = MinValue; j <= MaxValue; j++)
66.     {
67.         Amount += HistoGram[j]; //  像素总数
68.     }
69.
70.     PixelIntegral = 0;
71.     for (j = MinValue; j <= MaxValue; j++)
72.     {
73.         PixelIntegral += HistoGram[j] * j; //灰度值总数
74.     }
75.
76.     SigmaB = -1;
77.
78.     for (j = MinValue; j < MaxValue; j++)
79.     {
80.         PixelBack = PixelBack + HistoGram[j];
                                          //前景像素点数
81.               PixelFore  =  Amount  -  PixelBack;
    //背景像素点数
82.               OmegaBack  =  (double)PixelBack  /  Amount;
    //前景像素百分比
83.               OmegaFore  =  (double)PixelFore  /  Amount;
    //背景像素百分比
84.               PixelIntegralBack  +=  HistoGram[j]  *  j;
    //前景灰度值
85.      PixelIntegralFore  =  PixelIntegral  -  PixelIntegralBack;
    //背景灰度值
86.      MicroBack  =  (double)PixelIntegralBack  /  PixelBack;
    //前景灰度百分比
87.      MicroFore  =  (double)PixelIntegralFore  /  PixelFore;
    //背景灰度百分比
```

```
88.      Sigma  =  OmegaBack  *  OmegaFore  *  (MicroBack  -  MicroFore)  *
   (MicroBack  -  MicroFore);  //计算类间方差
89.      if  (Sigma      >      SigmaB)
   //遍历最大的类间方差 g      //找出最大类间方差以及对应的阈值
90.                 {
91.                       SigmaB      =      Sigma;
92.                       Threshold      =      j;
93.                 }
94.           }
95.           return      Threshold;      //返回最佳阈值;
96. }
```

2 转向控制策略

```
1. void     Turn_Cam_dias(void)
2. {
3.      float     temp;
4.      static     float     car_straight_dias_old;
5.
6.      car_straight_dias     =     M_Slope_fig()     *
   SERVO_DIVIDE_ANGLE_SCALE;
7.
8.      Straight_offset_filter();
9.      car_center_dias     =     car_center();
10.     Center_offset_filter();
11.     if(Road     ==     0     &&     Road0_flag     ==     0     &&
   Road0_flag0_flag     &&     fabs(car_straight_dias     -
   car_straight_dias_old)     <     30     )
12.     {
13.          temp     =     car_straight_dias     +
   PID_CAR_STRAIGHT_CAM.D     *     (car_straight_dias     -
   car_straight_dias_old);
14.     }
15.     else
16.     {
17.          temp     =     car_straight_dias;
18.     }
19.
20.     car_straight_dias_old     =     car_straight_dias;
21.     car_straight_dias     =     temp;
22.     if     (fabs(car_center_dias)     <     10)
23.     {
24.          car_center_dias     =     0;
25.     }
26. }
```