第十五届全国大学生
智能汽车竞赛

技 术 报 告

学　　校：　浙江大学

队伍名称：　浙大一队

参赛队员：　葛明阳 贾茗凯 刘科明

带队教师：　韩涛 姚维

目　录

第一章　总体方案

智能车的核心单元是 RT1064 单片机。i.MX RT1064 是 i.MX RT 系列芯片，是由 NXP 半导体公司推出的跨界处理器芯片，该系列下又包括 i.MX RT1020、i.MX RT1050 及 i.MX RT1060 等子系列芯片。所谓“跨界”，是指它自身的定位既非传统的应用处理器也非传统的微控制器。[1]

传统的应用处理器，如手机主控芯片，通常采用 ARM 的 Cortex-A 系列内核，配合其芯片架构使得芯片能实现更高频率的运行。传统的微控制器也称为 MCU，通常采用 ARM 的 Cortex-M 系列内核，相对来说该内核对中断响应更快，所以具有良好的实时性，但其芯片架构特别是集成片内闪存带来了生产技术限制和成本负担，从而限制了其性能。而 i.MX RT 系列芯片集成了两者的优点，其基于应用处理器的芯片架构，采用了微控制器的内核 Cortex-M7，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。

逐飞 RT1064 核心板，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。由于十五届全国大学生智能汽车竞赛中摄像头的高度从 20cm 下降到 10cm，所以对处理器的性能要求更高。传统的基于 OV7725 等二值化摄像头的开发方式虽然数据量小，易于单片机处理，但是已经不再满足新的赛道条件的需求，我们彻底弃用 7725 的方案，而采用 MT9V034 灰度摄像头，使用边缘检测与二值化并用的方式进行图像的预处理，而更复杂的算法需要更强的单片机性能，同时使用跨界处理器对嵌入式系统的学习也大有裨益，所以综合各方面考虑，本次比赛选择了 RT1064 方案，摄像

头采用总钻风 MT9V032，150FPS 下工作。高帧率带来的好处是图像更新速度更快，能够采集到的信息量更大，但缺点是图像亮度会受影响，且过快的帧率会超出控制的需求，即图像更新的信息在控制时可能不被采用，通过配置摄像头增益等参数可以对图像优化，需要根据实验效果做出取舍。

智能车系统的工作原理是：

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：

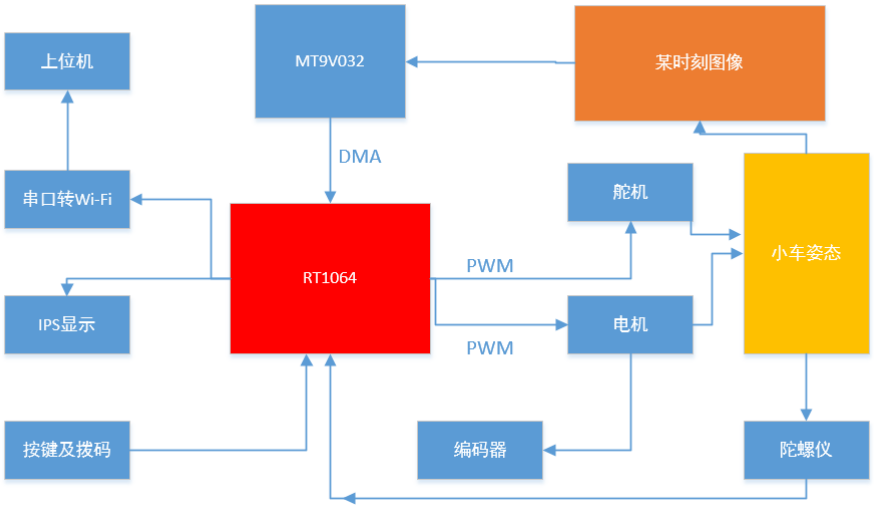

图 1 系统框图

第二章 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。

2.1 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。

2.2 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。

2.3 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。

第三章 电路设计

电路设计采用了主控板和驱动板分开的方式，这样一方面减少电机驱动部分电路对主控的干扰，另一方面缓解了布线压力，所有电路板都可以采用双面

板。之后我们对电路板的设计进行了优化，将主板以及电机驱动板均做成了 4 层 PCB，驱动板的面积缩减为不到原来的 1/2，主板的面积缩减为原来的 3/4。更小的面积有利于电路板的安装。

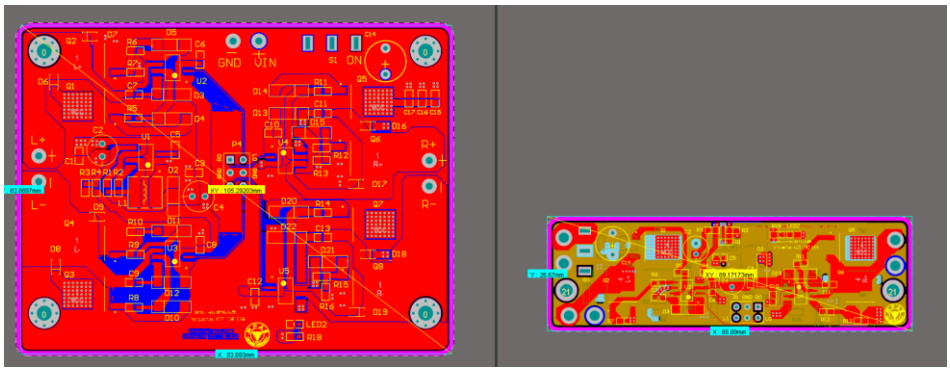

图 2 原驱动板（左）与新驱动板（右）

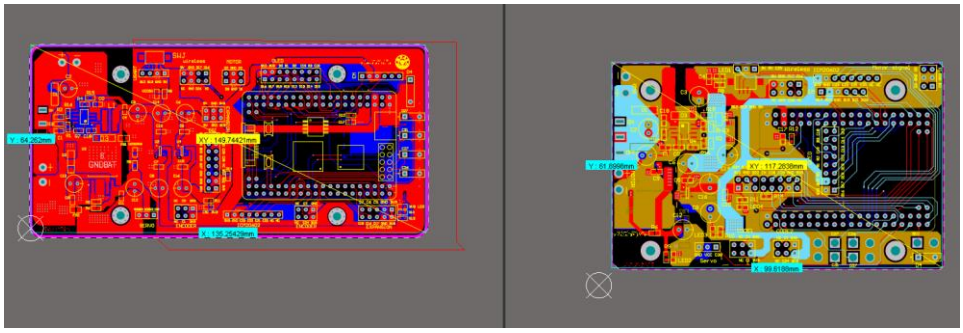

图 3 原主板（左）与新主板（右）

由于 RT1064 芯片外围电路较为复杂，主频较高，对电源质量、时钟信号走线都有很高要求。所以综合考虑下，相比绘制完整主板，直接采用官方核心板无疑是一个更好的方案。

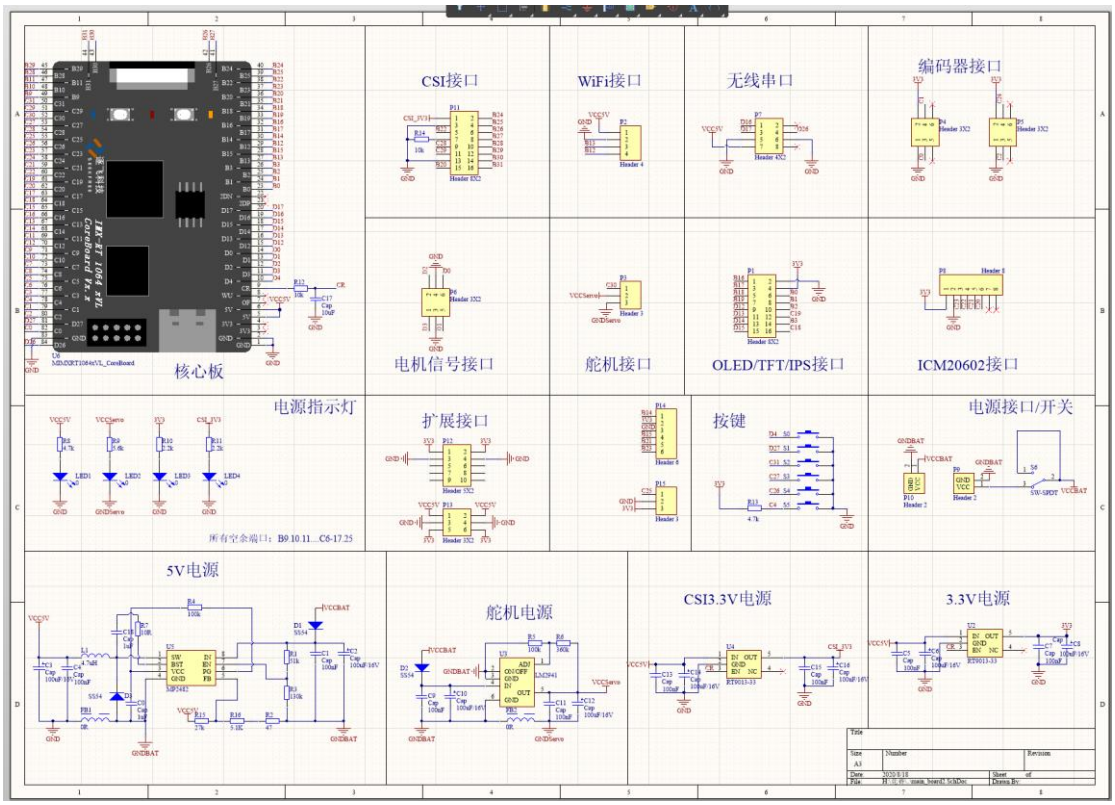

图 4 主板原理图

- **主板：**首先采用 MP2482 进行电源电压到 5V 的稳压，相比以往采用的 LDO，该芯片为 buck-boost，在电源电压被电机堵转拉低时，还有较好的稳压能力和输出能力，该芯片相较于 TPS63070 最大的优势在于封装为 SOC-8,非常易于焊接，而 63070 的封装为 VQFN，即无引脚封装，焊接起来存在一定困难。其输入电压最低为 4.5V，输入电压范围比 TPS63070 小，但已经可以基础四轮组对于电源的要求。
- **3.3V 稳压：**采用了两路 RT9013-33LDO 线性稳压器，一路供给摄像头，一路供给 IPS 屏幕等传感器。该稳压器电路十分简单，且易于焊接调试。
- **舵机电源：**采用了 LM2941 提供 6V 的电压，该方案能够提供的最大输出

电流为 1A，LM2941 的限流能力能够很好的保护舵机。

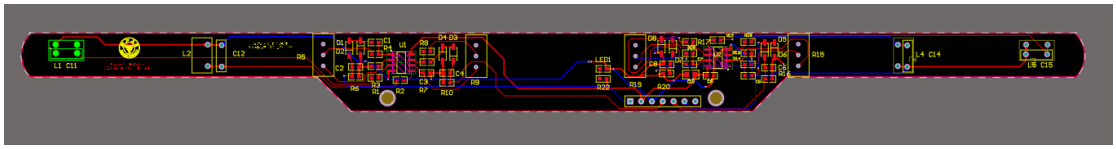

图 5 电磁杆

- **电磁运放电路：**选择 OPA 系列的运放，为节省空间将运放电路放置在电磁杆上，此设计可以节省主板空间。
- **驱动电路：**沿用最常用的方案，2104（驱动）+7843（MOS），性能稳定，驱动需要一个 12V，因此采用的是 34063，该升压电路的反馈电阻（0.22Ω）一定要保证阻值正确，且电容不能过大，否则电路工作不正常。另外，要保证通过大电流的通路的铺铜宽度足够。

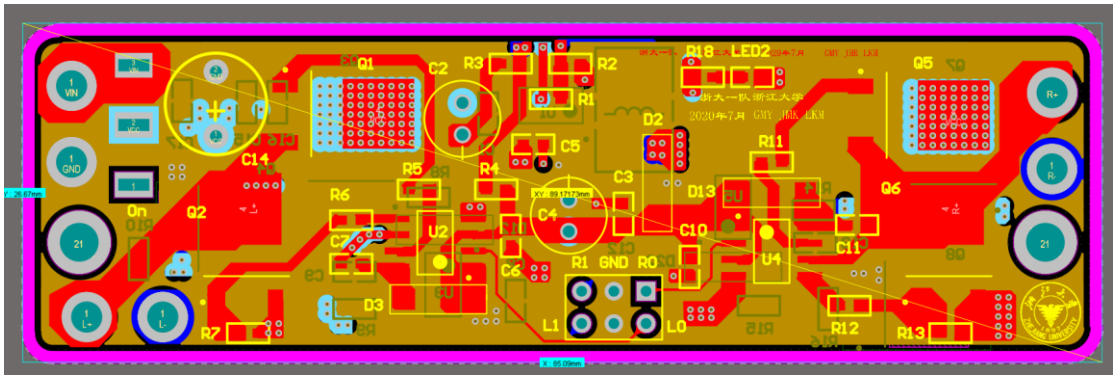

图 6 驱动板

电路设计以稳定、通用、成熟方案为宗旨，因为智能车电路不涉及过多的 PCB 设计知识，只要布局合理，电源处理恰当，即可工作正常，所以尽量不要追求过分个性化的设计，比如异形 PCB，过度密集的布局、过分追求更小等。

第四章 图像信息获取

4.1 数字摄像头的使用

摄像头采用逐飞的 MT9V032，通过一个 51 单片机配置，通过 MCU 的串口传给 51 数据即可进行配置，可以对帧率、自动曝光、增益等进行配置。经过实践，二值化图或 IPS 原图的图像质量均符合要求。

4.1.1 数字摄像头介绍

- MT9V032 的优势有：
  - 全局快门，相比卷帘快门，在高速下依然可以保持非常好的图像质量；
  - 高动态性能，90DB+的高动态范围。
  - FPS 可调，可自动曝光、曝光时间可调，不需要自己写驱动程序，也可通过取位的方式实现二值化。
- 图片大小、分辨率的选择

摄像头的大小和分辨率是 752*480，可以通过将两个或者四个像素点并成一个的方式，将图像进行降采样，以加快计算速度。经过裁剪，可以最终将图像变成 80*60 的图像，而视域基本不变，只需相应处理畸变、配合舵机控制程序进行一定调整，使用起来相比 7725 等固定分辨率相机更加灵活

- 采用 140 度无畸变广角镜头

由于环岛等赛道元素的存在，且摄像头高度逐年的降低，因此采用广角的摄像头获取更宽的视野。但更宽的摄像头也存在图像过远和入弯补线错误等情况，需要同时调整高度和程序，以达到理想的效果。

4.1.2 数字摄像头的使用

数字摄像头用 SCCB 协议对摄像头的寄存器进行配置，单片机需要配置好中断、DMA 传输，自动获取摄像头图像。

1) 摄像头硬件连接

9V032 摄像头模块是 3.3V 供电，不能和 MCU 等用一个 3.3V 稳压芯片供电，应单独使用一路 3.3V LDO 进行供电。除了 3.3V 和 GND，还有行场中断，串口（一般摄像头为 SCCB），8 位数据线。

2) SCCB 寄存器配置

SCCB(Serial Camera Control Bus)是和 I2C 相同的协议，一般由两根线组成，一根 SCL 时钟线和一根 SDA 数据线。在摄像头初始化的时，可以用 SCCB 协议对摄像头的寄存器进行配置。

3) 中断传输

摄像头一般有两种传输方式，一种是场中断传输方式，一种是行中断传输方式，场中断即场中断信号来后开启 DMA 传输，直到一幅图传输完成；行中断即行中断信号来后开启 DMA 传输，有几行就需要进几次行中断。在初始化时需要配置好行/场的中断源、触发边沿和上/下拉电阻。然后在中断函数中清空标志位和打开 DMA 传输。

4) DMA 配置

数字摄像头可使用 DMA（Direct Memory Access）技术。DMA 是一种直接存储器访问技术，工作过程中不需要 CPU 干预，也不需要像中断处理方式那样需要保留现场、恢复现场。

若关闭 DMA，则需要占用 CPU 进程用于摄像头数据传输，无法执行主循环程序。开启 DMA 后，摄像头有数据过来，CPU 开启 DMA 记录摄像头数据，在 DMA 记录数据时，CPU 可以执行其他的任务，等 DMA 记录完数据后，CPU 再对数据进行图像处理，节约了时间。

4.2 图像二值化阈值选取

图像二值化有很多算法。这里主要讲述大津法及平均值法两个方法

a) 大津法(OTSU)

大津法又名最大类间差方法，是日本学者大津于 1979 年提出的一种自适应阈值分割法。它通过统计整个图像的直方图特性，使得阈值T_0'能将图像分为相差最大的两类。

设阈值T_0'将图像分割为目标图像和背景图像两个部分，属于目标图像的像素点数占整幅图像的比例为ω_0，其平均灰度为μ_0；属于背景图像的像素点占整幅图像的比例为ω_1，其平均灰度为μ_1。图像总平均灰度为μ，类间方差为g。则有

$$\mu = \omega_0\mu_0 + \omega_1\mu_1 \qquad \text{(公式 1)}$$

$$g = \omega_0(\mu_0 - \mu)^2 + \omega_1(\mu_1 - \mu)^2 \quad \text{(公式 2)}$$

其中，$\omega_0 + \omega_1 = 1$。将(公式 1)代入(公式 2)有

$$g = \omega_0\omega_1(\mu_0 - \mu_1)^2 \quad \text{(公式 3)}$$

使得类间方差g最大的阈值T_0'即为最佳分割阈值。

b) 平均值法

平均值法是指将算整场图像像素的灰度值的平均值μ作为图像的分割阈值T_0'。[1]

由于单片机主频高，算力足，这里采用效果更好的大津法来算去全局阈值。然而由于图像四周受畸变影响较重，图像相对正常图像较暗，加之工程实际与理论存在一定差别，为了给图像阈值的更多可调整空间，实际选取阈值还需在计算结果上加上一定偏置，即$T_0 = T_0' + T_{offset}$。

4.3 反光点去除

由于赛道路面不完全平整，在灯光照射下会使得采集到的图像出现光强过高的“反光点”（即灰度值偏大），从而导致 2.1.1 中计算得到的阈值偏低。同时，此情况下也会影响后续边缘检测得到的赛道边线特征。

为解决此问题，我们对灰度图像进行了反光点去除处理。对于统计得到的图像直方图，可以获取每个灰度值 i（i=0,1,2,…,255）对应的像素点个数 N_i。由于反光点个数相对较少，故可以通过设置像素点个数阈值 N_{min} 来滤除该类型

噪点，从而获取更为准确的灰度图像。

$$N_i = \begin{cases} N_i, N_i \geq N_{min} \\ 0, N_i < N_{min} \end{cases} \quad \text{(公式 4)}$$

然后将被滤除的噪点灰度值置为与其余像素点最大灰度值相同，并更新灰度直方图用于阈值选取，从而实现反光点的去除。

4.4 边缘检测

在赛道光线不均匀时，仍有使得图像不稳定的情况，采用边缘检测的方式保证赛道边线提取的正确性就非常有必要。因此采用 Sobel 算子进行边缘检测。

索贝尔算子（Sobel operator）主要用作边缘检测，是一离散性差分算子，用来运算图像亮度函数的灰度之近似值。分别用横向和纵向 sobel 算子对灰度图进行卷积运算，可以得到图像灰度矢量信息和其法矢量信息。[2]

sobel 算子有横向和纵向两种卷积核：

$$S_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}, \; S_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad \text{(公式 5)}$$

将横向与纵向算子分别与图像平面卷积，得到横向及纵向的亮度差分值G_x及G_y。记灰度图像矩阵为 A，则有

$$G_x = S_x * A \;, \; G_y = S_y * A \quad \text{(公式 6)}$$

于是得到图像梯度矢量大小为

$$G = \sqrt{G_x^2 + G_y^2} \quad \text{(公式 7)}$$

由于赛道边缘点附近黑白梯度最大，故可认为大于临界阈值G_0的点为赛道边缘点。

由于摄像头透视效果的影响，图像远景区域会将更多的图像信息投射到同一个像素点，这也就会造成图像远景区域的赛道边缘临界阈值更小，故临界阈值G_0应在图像上分段或动态给出。

4.5 图像二值化

依据上述阈值选取结果与边缘检测结果，对灰度图像进行二值化处理。

对于灰度图像中每一个像素点，首先根据灰度值与阈值对比进行预二值化，对于上述的阈值T_0，灰度图像中每个像素点(x,y)，以及其对应的灰度值$\mathrm{i}_{(\mathrm{x},\mathrm{y})}$，记二值图像对应值为$\mathrm{p}_{(\mathrm{x},\mathrm{y})}$，则有：

$$\mathrm{p}_{(\mathrm{x},\mathrm{y})} = \begin{cases} 0\,, \mathrm{i}_{(\mathrm{x},\mathrm{y})} < T_0 \\ 1\,, \mathrm{i}_{(\mathrm{x},\mathrm{y})} \geq T_0 \end{cases} \quad \text{(公式 8)}$$

然后根据边缘检测结果，对二值图像结果为 1 的点（亮点）进行进一步检测，记每个像素点对应灰度梯度矢量大小为$\mathrm{G}_{(\mathrm{x},\mathrm{y})}$，则有：

$$\mathrm{p}_{(\mathrm{x},\mathrm{y})} = \begin{cases} 0\,, \mathrm{G}_{(\mathrm{x},\mathrm{y})} > G_0 \\ 1\,, \mathrm{G}_{(\mathrm{x},\mathrm{y})} \leq G_0 \end{cases} \quad \text{(公式 9)}$$

最终得到更新的二值化图像$\mathrm{p}_{(\mathrm{x},\mathrm{y})}$，并用于后续图像处理等。

4.6 边线识别

根据得到的二值化图像，检测黑白的跳变点以期望找到每一行的赛道边线位置，为了克服噪音的影响，采用不同严格度的边线检测方法。

当初步搜索的边线与前一行边线连续，且符合赛道畸变特征，则可以放宽判定条件，以避免条件过于严苛而丢失边线。否则，需要逐渐使用更加严格的条件以避免将噪音误判为边线情况的发生。

而对于圆环，在进入圆环之前的时刻。摄像头能同时看到前进方向的赛道及进圆环方向的赛道，故也需要对边线进行重搜，以保证搜到的边线是正确的路径。

4.7 图像校正

由于摄像头本身存在外畸变，故对图像做逆透视变换来得到小车坐标下的标准图像。

透视现象实际上是世界坐标到摄像头坐标的转换，即经过旋转、平移及投影变换，将每个点在小车坐标系的相对位置转换到摄像头坐标的相对位置，最后再经过一个投影，将摄像头坐标转换成平面坐标。由坐标变换的相关知识可以知道，坐标系的旋转、平移变换都可以用一个 3*3 的非奇异常数矩阵来表示，而投影可以用一个 3*3 的奇异矩阵表示。此外，由于智能车看到的图像都是平面的赛道，即 Zg= 0，故可得到如下方程：

$$\begin{cases} x_c = \dfrac{(d\cos\theta + h)X_G}{Y_G\sin\theta + d\cos\theta} \\ y_c = \dfrac{(d\cos\theta + h)Y_G - hd\sin\theta}{Y_G\sin\theta + d\cos\theta} \end{cases} \quad \text{(公式 10)}$$

其中X_G，Y_G是世界坐标系，x_c，y_c是摄像头坐标系。而求解该方程主要有三个参数：焦距 d，摄像头高度 h，摄像头俯仰角θ。

求解逆透视主要有 3 个参数，由于理论计算、测量得到的参数精度不够，所以采用系统辨识的方式得到关键参数是一个很好的选择。

由于对整幅图像进行逆透视计算量较大，所以只对关键信息——边线进行逆透视是一个更好的选择。于是通过逆透视矫正，将赛道矫正为宽为 300 像素的标准小车坐标下的赛道。

第五章 电磁信息获取

5.1 理论模型

由于电磁传感器放置位置受限于摄像头视野范围，仅能达到 3-5cm 左右的前瞻。为矫正传统偏差计算方法带来的误差，改进经典电磁模型以适配传感器状态。

首先一根无限长通电导线的磁感应强度与导线距离的平方成反比：

$$\mathrm{B} = \frac{\mathrm{B_0}}{r^2} \tag{公式 11}$$

赛道电磁线在小尺度下可看作无限长通电导线。且其中通交流电，所以通过电感的磁感应强度及对应感生电流也是交变的。记电感到电磁线距离为 l，则有感生电流的大小$\mathrm{I_{感应}}$：

$$\mathrm{I_{感应}} \propto 1/l^2 \tag{公式 12}$$

使用距离平方的反比代替小车测得的电感电压。实际系统中将交变电流做了交-直变换后得到了电感电压，由于变换线性，故测量值仍与距离平方反比成正比例关系。

记电感距离地面电磁线的高度为 h，左右电感相对距离为 n。则有两侧电感测量值$\mathrm{u_{测量\,1}}$、$\mathrm{u_{测量\,2}}$对应表达式：

$$\begin{cases} \mathrm{u_{测量\,1}} \propto 1/(h^2 + a^2) \\ \mathrm{u_{测量\,2}} \propto 1/(h^2 + b^2) \\ a + b = n \end{cases} \tag{公式 13}$$

以上公式未考虑电感与电磁线非完全垂直的情况，受电感安装位置影响，其更易随车头而左右晃动，使得测量值出现较大波动。此时需要针对角度量执行矫正，减小测量误差。

在引入角度偏差的情况下，上述公式修正为[3]：

$$\begin{cases} \mathrm{u}_{测量\,1} \propto \frac{1}{h^2+a^2} * \cos(\theta) \\ \mathrm{u}_{测量\,2} \propto \frac{1}{h^2+b^2} * \cos(\theta) \\ a+b=n \end{cases} \quad \text{(公式 14)}$$

其中 θ 代表电磁线偏离小车电磁感法向量的角度。

假设在测量点引入另一个电感，该电感垂直于原电感，则同理有电感测量值公式：

$$\begin{cases} \mathrm{u}_{测量\,1}{}' \propto \frac{1}{h^2+a^2} * \sin(\theta) \\ \mathrm{u}_{测量\,2}{}' \propto \frac{1}{h^2+b^2} * \sin(\theta) \\ a+b=n \end{cases} \quad \text{(公式 15)}$$

因此理论上只需要将两个测量值相除直接可得 θ 的正切值，但是实际系统中，由于并不是理想的无限长直导线，以及测量值较小时存在较大测量误差，因此应用中选择测量值相对大的一组电感相除来计算 θ 角。由此我们可以得到该偏差角。此外，如果假设车辆轨迹是理想的电磁线切线，该角度反馈为弯道曲率，可用于前文所述主动差速的输入。

因为在上述方程组中仅有 a 和 b 是未知量，而求取$\frac{(a-b)}{2}$也就是中线偏差是主要目的，所以可以直接得到这个偏差正比于一个量：

$$\frac{(a-b)}{2} \propto \left(\frac{1}{\mathrm{u}_{测量\,1}} - \frac{1}{\mathrm{u}_{测量\,2}} \right) * \frac{\cos(\theta)}{n} \quad \text{(公式 16)}$$

再结合前述 θ 表达式：

$$\cot(\theta)=\begin{cases}\dfrac{u_{测量1}}{u_{测量1}'} & u_{测量1}{}^{2}+u_{测量1}{}'^{2}>u_{测量2}{}^{2}+u_{测量2}{}'^{2}\\ \dfrac{u_{测量2}}{u_{测量2}'} & u_{测量1}{}^{2}+u_{测量1}{}'^{2}<u_{测量2}{}^{2}+u_{测量2}{}'^{2}\end{cases} \quad \text{(公式 17)}$$

即可求得中线偏差从而给后级舵机 PD 控制使用。

上述模型中依然存在一些误差：首先两个放置很近的电感存在无法抵消的互感现象，对称布局三个电感、使用铝箔电磁屏蔽等方法对解决该问题效果不佳；另一方面由于电感体积较大，两个电感并非在同一点测量场强，造成测量误差。

5.2 系统辨识

在实际应用中，利用左右电感的单侧或双侧值，可以得到小车中心线距赛道重心线的距离。根据理论推导可知，小车电感的采样值与小车与中心线距离成反向单调趋势，故可以通过分段线性拟合的方式拟合出小车电感值与小车中心线距赛道中心线的距离。

由于两电感安装位置有一定距离，故两个电感不会完全饱和。因此，根据单侧值或双侧值可以很容易拿到小车的位置。

然而，此种方式在赛道前方有弯道、十字、圆环等赛道类型时，该种方法并不适用。但在短坡道上采用控制小车向前行进，电磁导航是足够稳定的

5.3 电磁保护

小车在高速行进过程中，遇到撞击、翻车等事故对小车的机械结构及硬件

设备都会带来很大的损坏，而赛道外无电磁的故有故有特征可以用来做小车冲出赛道的判断条件，以及时刹停小车。但也要防止小车飞离赛道或车较偏时的误判定。

第六章 方向控制

小车方向控制由两个部分构成：其一是使得小车行进方向与赛道切线方向平行（以下简称车直），保证小车沿赛道前进；其二，即使小车沿着赛道行进，若其与赛道中心线位置有所偏移，会影响小车后续状态的稳定性，故需要控制舵机使得小车向中间修正，使得小车处于赛道的正中间（以下简称车正）。

控制行，即用来计算转向偏差的参考行，表征用于方向控制的图像行数。速度较快时，控制行需要选取较远行，从而保证舵机角度控制提前。由于舵机额定频率为 50Hz，舵机响应的滞后需要用前瞻来进行补偿。

6.1 车直转向偏差值的计算

根据阿克曼转向模型，小车行进曲率半径等于前方中心线曲率半径的转角，可以控制角度使得小车前轮平均打角应与中心线切线平行。

由于舵机打角滞后、中心线曲率拟合准确率低或计算量大、小车车轮与地面存在打滑等因素。所以，为了快速且准确的计算舵机打角，可以考虑将中心线的斜率做为车直偏差。

通过最小二乘法计算出中心线斜率，在入弯、弯心、出弯等过程中，与小车前轮需要的打角之间呈正相关的，因此该斜率可作为车直转向偏差。

6.2 车正转向偏差值的计算

车正转向偏差，即近景行中心线与小车中心位置的偏差。由于小车在转弯

时赛道中心线与小车中心位置存在固有偏差，因此在转弯时，对固有偏差做偏置处理，以减小误差。

6.3 车直舵机的 PD 控制

由于车直偏差与舵机打角存在线性关系，所以直接采用比例控制。由于小车与地面存在打滑现象，所以当速度增加时，舵机打角也需要增大，为了使小车更稳定，我们通过将 P 参数和速度进行耦合，解决了不同速度下舵机打角不同的问题，且调试方便。该方法极大的提升了小车变速行驶时的稳定性。

由于纯比例控制会导致小车在直路上超调较大，故通过微分作用消除超调及震荡。

6.4 舵机的 PD 控制

由于车正偏差与舵机打角为非线性关系，所以直接采用 PD 控制效果不佳。而将非线性偏差映射到线性偏差，也是困难且不必要的。所以，采用模糊 PD 控制，设置线性 PD 参数表，并和速度进行耦合，来使得小车快速且无超调的恢复到赛道中心位置。

第七章 速度控制

7.1 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。

7.2 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。

7.3 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。

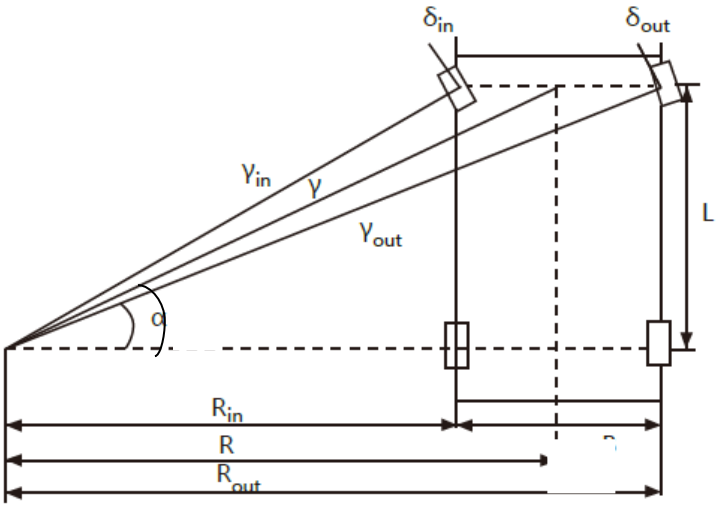

图 7 阿克曼转向模型

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

$$\omega = \frac{v}{R} \qquad \text{(公式 18)}$$

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

$$v_1 = \omega R_{in} \tag{公式 19}$$

$$v_2 = \omega R_{out} \tag{公式 20}$$

由阿克曼转向模型几何关系可知

$$R = \frac{L}{\tan\theta} \tag{公式 21}$$

于是有两轮差速

$$v_2 - v_1 = v\frac{W}{L}\tan\theta \tag{公式 22}$$

也即

$$\begin{cases} v_1 = v\left(1 - \dfrac{W}{2L}\tan\theta\right) \\ v_2 = v\left(1 + \dfrac{W}{2L}\tan\theta\right) \end{cases} \tag{公式 23}$$

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

$$\begin{cases} v_1 = v\left(1 - K_p \dfrac{W}{2L}\tan\theta\right) \\ v_2 = v\left(1 + K_p \dfrac{W}{2L}\tan\theta\right) \end{cases} \tag{公式 24}$$

第八章 其他传感器

陀螺仪在四轮光电组中速度控制方面并不是很重要，但是在上坡识别中，可以采用图像、激光和陀螺仪的方法，其中采用图像较为麻烦，采用激光又存在误判的情况，且在上坡过程中，采用陀螺仪积分的方式配合电磁打角比其他方式更加具有可控性，所以采用陀螺仪是非常有必要的。

我们选用的数字式陀螺仪 ICM20602，它支持 I2C 和 SPI 接口，可采用库中的硬件 SPI 配置和采集方式，数据稳定。事实证明 ICM20602 比常见的 MPU6050 性能稍好。同时，我们还采用了卡尔曼滤波[4]，对裸数据进行了标定，实际数据准确，上坡检测效果好。

陀螺仪判定坡道会存在两个问题，第一个问题是小车压到路肩时会出现误判，故还需加入误判检测，这里不再详细赘述；另一个问题是陀螺仪判定坡道时需要对角度进行判定，梯形坡道和圆弧坡道的特征不同，也需要根据实际情况对参数进行修改。

第九章 开发工具及调试过程

开发工具选用 IAR (IAR Embedded Workbench)开发环境，它是一款应用于嵌入式系统的开发工具、C/C++的编译环境和调试器。其特征在于高效的 PROMable 代码；完全标准 C 兼容；内建对应芯片的程序速度和大小优化器；便捷的中断处理和模拟；以及高效浮点支持等。通过 JLink 等硬件仿真器，可以实现在线仿真调试。

硬件仿真器采用 DAP 下载器。DAPLINK 是一个可以在 ARM CORTEM MCU 上进行编程和调试的测试器，其具有 CDC 虚拟串口功能，可以在调试的同时进行串口通信；同时兼容两种下载调试模式：HID 免驱动兼容模式，WINUSB 高速模式。

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

```
1.  uint8_t GetOSTU(void)
2.  {
3.      int16_t i, j;
4.      uint32_t Amount = 0;
5.      uint32_t PixelBack = 0;
6.      uint32_t PixelIntegralBack = 0;
7.      uint32_t PixelIntegral = 0;
8.      int32_t PixelIntegralFore = 0;
9.      int32_t PixelFore = 0;
10.     double OmegaBack, OmegaFore, MicroBack, MicroFore, SigmaB, Sigma; // 类
    间方差;
11.     int16_t MinValue, MaxValue;
12.     uint8_t Threshold = 0;
13.     uint16 HistoGram[256]; //
14.     int16_t Mincount = 0, Maxcount = 0;
15.
16.     for (j = 0; j < 256; j++)
17.     {
18.         HistoGram[j] = 0; //初始化灰度直方图
19.     }
20.
21.     for (j = START_LINE; j < CAMERA_H; j++)
22.     {
23.         for (i = 0; i < CAMERA_W; i++)
24.         {
25.             HistoGram[Image_Use[j][i]]++; //统计灰度级中每个像素在整幅图像中的
    个数
26.         }
27.     }
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_d
    ias - 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. }
```

二、 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。