东北大学秦皇岛分校

Northeastern University at Qinhuangdao

毕业论文

智能交通半实物仿真平台的实践

院　　别	控制工程学院
专业名称	自动化
班级学号	**20178061**
班级序号	**170205**
学生姓名	徐寅达
指导教师	郭戈

2021 年 5 月 31 日

智能交通半实物仿真平台的实践

摘　　要

在现代社会，智能交通系统是解决当今交通问题的有效方法，为推动智能交通系统而产生的智能交通仿真平台也需要得到发展。由于实车模拟的成本过大且许多交通场景难以复现，目前主流的智能交通仿真平台仍以软件仿真为主。但是软件仿真也有其局限性，例如模拟真实情况困难。而智能交通半实物仿真平台可以在软件仿真和实车模拟的优缺点中各有所取舍，在智能交通系统的研究中起到更佳的作用。

本文主要介绍了小车的各功能模块。首先，根据需求选择了 Arduino 作为智能小车的主控模块。小车的各项功能主要通过电磁循迹模、IC 卡识别模块 WiFi 无线串口模块以及 OpenMV 摄像头模块组成。借助电磁感应原理小车能够在车行道上平稳行驶而不至于驶出车道。通过 IC 卡识别模块小车可以获取自己的位置信息。除此之外，本文建立了基于无线 WiFi 串口模块的通信网络来实现小车与上位机的通信。结合 IC 卡识别模块和通信网络，小车不仅能够向上位机汇报自各自的位置，还能够通过上位机发来的规划信息来达到路径规划的功能。借助于 OpenMV 平台，小车也能够对前方的交通信号灯进行识别并且做出正确的行驶决策，对于前方的其他车辆，小车也能够保持安全的车距。最后，本设计达到了基本的预期效果，小车可以在平台上一起行驶，并根据各自的路径规划信息进入各自的车库。

关键词：Arduino，智能交通系统仿真平台，OpenMV，IC 卡

Practice of hardware in the loop simulation platform for Intelligent Transportation

Abstract

Intelligent transportation system (ITS) is an effective method to solve modern traffic problems. In order to develop ITS, the simulation platform of ITS also needs to be developed. Because the cost of real vehicle simulation is too large and many traffic scenes are difficult to simulate, the mainstream intelligent transportation simulation platform is still based on software simulation. But software simulation also has its limitations, for example, it is difficult to simulate the real situation. The hardware in the loop simulation platform of intelligent transportation can choose between the advantages and disadvantages of software simulation and real vehicle simulation, which plays a better role in the research of intelligent transportation system.

This paper mainly introduces the function modules of the car. Firstly, Arduino is selected as the main control module of the intelligent car. The functions of the car are mainly composed of electromagnetic tracking module, IC card identification module, WiFi wireless serial port module and OpenMV camera module. With the help of the principle of electromagnetic induction, the car can run smoothly on the roadway without leaving the lane. Through the IC card identification module, the car can obtain its own location information. In addition, this paper establishes a communication network based on wireless WiFi serial module to realize the communication between the car and the host computer. Combined with IC card identification module and communication network, the car can not only report its own position to the upper computer, but also achieve the function of path planning through the planning information sent by the upper computer. With the help of OpenMV platform, the car can also identify the traffic lights in front and make correct driving decisions. For other vehicles in front, the car can also keep a safe distance. Finally, the design achieves the basic expected effect, the car can drive together on the platform, and according to their own path planning information into their garages.

Key Words: Arduino, intelligent transportation Platform, OpenMV, IC card

目　　录

1 绪论

1.1 研究背景和研究意义

交通是近现代人类生活生产，社会进步的关键因素。随着社会的发展，人类对交通系统提出了更高的要求。当今社会，交通系统可以主要归为四类，第一类是以空中客机为主的空中交通，第二类是以机动车为主的道路交通，第三类是以火车地铁为主的轨道交通，第四类是以轮船为主的水上交通。他们的发展为人类提供了便利、优质的交通体验。但是，由于各国人口数量的爆炸式增长，城市化的发展，各国交通道路的建设、升级速度一直都滞后于汽车数量的增长速度。交通堵塞的问题越来越常见也越来越严重，大大制约了社会生产效率的提高。同时，由于拥堵，车辆能源的利用效率也大打折扣，进而也造成了大气环境的污染[1]。幸运的是，人们开始察觉到智能交通系统（ITS）的发明和快速发展，能够有效地提高城市道路的使用效率进而解决交通拥堵带来的一连串问题。实践证明，智能交通系统（ITS）是解决目前经济建设发展所带来的交通问题的最优解决方法[2-4]。

一直以来智能交通系统的研究主要是采用软件仿真的方法，但是这个方法存在模拟真实情况困难的问题。然而实车仿真的方法也有很多局限性，例如实车的成本过于昂贵，许多实际路况无法完全配合测试所需，交通安全难以保证，难以构造出许多以外情境。智能交通半实物仿真平台是旨在满足智能车辆演示操作需求的半实物仿真平台，可以模拟出各种各样的交通场景，能够降低智能交通系统设计研发的周期和成本，并在系统设计的安全性和便利性方面有明显的优势[5]。

1.2 国内外研究现状

1.2.1 国外研究现状

智能交通半实物仿真平台由场景沙盘、信号灯、车辆管理端、智能小车和通信网络构成。沙盘及信号灯为智能小车模拟出了真实驾驶场景[6-7]，而车辆管理端通过通信网络与智能小车建立了联系。通过车辆管理端和沙盘信号灯所给的信息，小车能在沙盘上按规划行驶。

日本的 Forum8 公司已经在这方面有了比较深层次的研究，并将研究转化为产品在进行销售。他们研发了具有驾驶模拟功能的虚拟现实软件 UC-win/Road 以及和车辆机器

人平台＜Robocar＞连接的 VR 模拟系统。而这 VR 模拟系统功能强大，它可以在 VR 空间的道路上通过控制实车十分之一规模的 scale model 车辆来实现仿真目的[8]。

利用 VR 技术表现的虚拟现实可设置精致的空间、丰富的交通环境以及脚本并进行试乘，是可以广泛应用于车辆机器人的研究开发、尖端安全汽车以及 ITS 的研究开发的系统。图 1.1，图 1.2 展示了他们 VR 模拟场景的实现。

图 1.1 在办公室行驶的 Robocar

图 1.2 制作模型等行驶环境后可在 VR 空间行驶

1.2.2 国内研究现状

在智能交通半实物仿真平台的研究方面，我国已经有一些单位和个人进行过研究。毕业于上海交通大学的硕士研究生陆正辰在他的硕士毕业论文《基于多缩微车的智能交通系统仿真平台研究》[9]一文中展现出了他们的实验室在这方面以及有很高的研究水平。提出了高、低成本的智能车硬件方案，对智能车的可行驶区域检测、障碍检测、交通信号灯检测、地面标志检测与识别方法进行了测试和总结分析。除此之外，他还对缩微智能车变道保持、路口控制、泊车策略及控制进行了深入的研究以及实践论证。

此外赵津前辈在他的《基于城市先进道路交通系统的智能交通仿真平台设计》[10]一文中也展示了他们设计的智能交通半实物仿真平台。他更有目的的对平台进行了多种区域的划分，可以针对各种场景进行实验。并且他们基于 arduino 和 x86 工控机的硬件

系统也非常高效。丰富的传感器使该微缩智能车能够进行例如车辆循迹、障碍物检测交通信号灯检测等任务，还可以进行全局协同控制与局部路径规划，实现纠正偏离车道的小车、躲避障碍物等功能。

学者苏致远也在《基于多缩微车的智能交通半实物仿真平台》[11]一文中提出了他的平台方案设计。首先是车载系统的设计，主要分为道路模式识别以及障碍检测。小车通过各传感器，能够判断车辆目前处在上坡、下坡还是平坦道路行驶状态，并能够在检测完障碍物后，进行超车等操作。然后他还解释了在它的设计中车队协同驾驶系统设计：由协作层的车队协同驾驶混成自动机、控制层的车载硬件平台和道路交通环境构成的多缩微智能车的协同驾驶系统。最后，他在文中还提出了基于相机以及雷达的同时定位与地图创建（SLAM）技术，并对此方向的研究提出了憧憬。

1.3 论文的基本内容及拟采用的研究方法

智能交通仿真平台信息传递结构示意图见图 1.3。PC 上位机通过 WiFi 无线网络向智能小车和仿真沙盘以及交通信号灯传递信息，传递的信息分别是小车的控制信号和交通信号灯的控制信号。小车的各项功能主要通过电磁循迹模、IC 卡识别模块 WiFi 无线串口模块以及 OpenMV 摄像头模块组成。借助电磁感应原理小车能够在车行道上平稳行驶而不至于驶出车道。通过 IC 卡识别模块小车可以获取自己的位置信息。除此之外，本文建立了基于无线 WiFi 串口模块的通信网络来实现小车与上位机的通信。结合 IC 卡识别模块和通信网络，小车不仅能够向上位机汇报各自的位置，还能够通过上位机发来的规划信息来达到路径规划的功能。借助于 OpenMV 平台，小车也能够对前方的交通信号灯进行识别并且做出正确的行驶决策，对于前方的其他车辆，小车也能够保持安全的车距。

本文设计并实现的缩微智能场景如图 1.4 以及图 1.5。

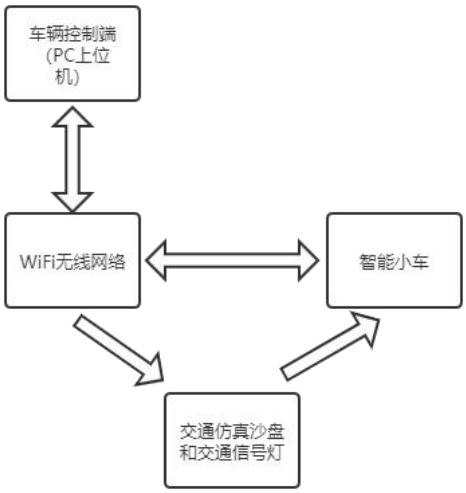


图 1.3 智能交通仿真平台信息传递结构图

图 1.4 缩微交通场景一

图 1.5 缩微交通场景二

1.4 论文的主要章节结构

主要分为七个章节：小车主控方案设计、小车循迹方案设计、小车程序无线下载模块设计、通信网络模块分析与设计、位置信息以及路径规划功能设计、交通信号灯识别功能设计以及避障功能设计。每一章节主要就是与其他可能可行方案的对比以及所选择方案的软件设计和硬件设计。本文的主要章节结构图如下图1.6。

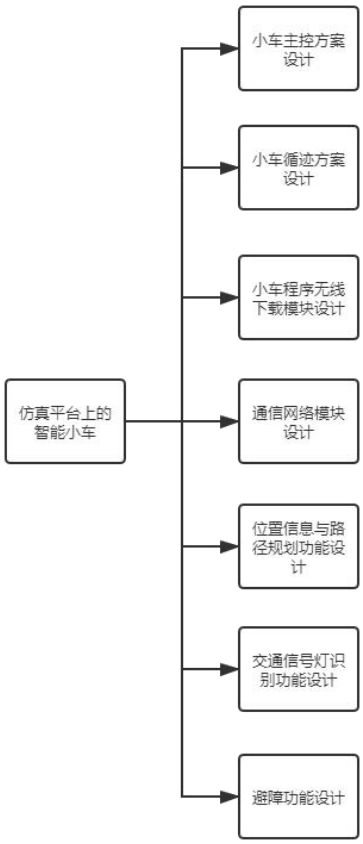


图1.6 本文的主要章节结构

2 小车主控方案设计

2.1 方案设计

主要的方案就两个：1.嵌入式微处理器（MCU）方案。2.单板工控机方案。这两个方案的区别是：

（1）功耗差距大，单板工控机的功耗比嵌入式微处理器的功耗大很多，因此，对于电源会有更高的要求。对于电源更高的要求，就会受制于成本和尺寸因素。

（2）尺寸差距大，单板工控机的尺寸比嵌入式微处理器大很多，除此之外，由于单板工控机的功耗需求也大，因此响应的电源部分的尺寸肯定也会比嵌入式微处理器的电源部分的尺寸大很大。尺寸的大小是受制于车道宽度的，智能小车的宽度不能宽到妨碍小车在车道正常行驶的程度。

（3）算力差距大，单板工控机的计算能力比嵌入式微处理器高很多，涉及到图像处理，目标检测等的内容，单板工控机有非常大的优势。

（4）成本差距大，单板工控机的价格相较嵌入式微处理器的价格昂贵不少，如果需要大批量的使用，成本将会非常高。

考虑到本半实物仿真平台需要用到大量的小车，如果全用单板工控机，成本会过于高，而嵌入式微处理器方案的成本会低很多。因此最后决定第一个方案，用Arduino Mega 2560单片机作为本设计的主控模块。Arduino Mega 2560是基于ATmega2560芯片的微处理器控制板，串口接口非常丰富，它还能通过USB接口直连电脑，足以满足本设计的大部分需求。Arduino Mega 2560的电路原理图如图2.1。

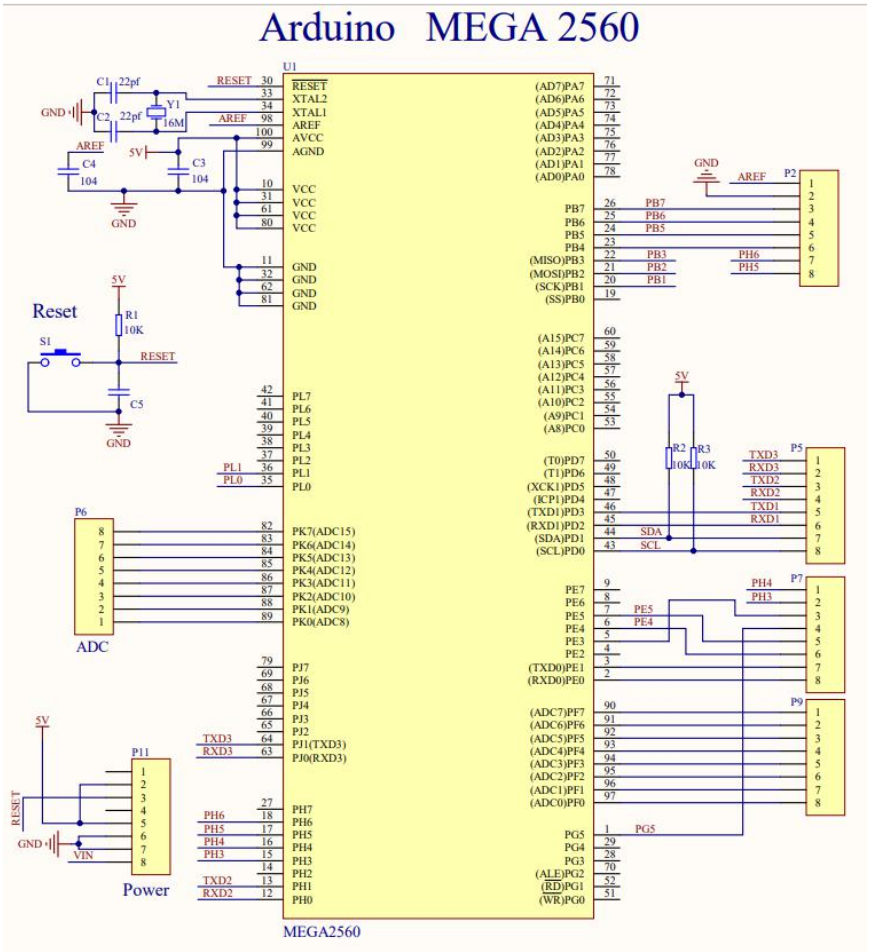


图 2.1 Arduino Mega 2560 的电路原理图

2.2 本章小结

本章主要介绍了小车主控方案的选择讨论，主要讨论了嵌入式微处理器（MCU）方案和单板工控机方案的优缺点。并且介绍了所采用的嵌入式微处理器（MCU）Arduino Mega 2560的硬件信息和电路原理图。

3 小车循迹方案设计

3.1 小车循迹方案的选择

目前主流的寻迹方案大概有三种：电磁寻迹寻迹，红外光电寻迹，摄像头寻迹。

红外光电寻迹的方案一般是采用至少两个对反射式红外对管，将其固定在小车的顶部前端。红外对管会发射光信号，并接受反射回来的光信号，根据反射面属性以及距离的不同，反射回来的光信号也会不同。通过测量被测信息上的光波并进行调制，通过空间分布、相位、强度等的变化，将光信号转化成电信号，再将电信号进行调制分离得到可用的信号。简单地来说，就是红外光对于不同颜色的反射面会反射不同强度和性质的红外光。接收管接受反射回来的红外信号后会将其转化成电信号的形式，比如电压。根据所测得的红外模块电压值的变化，可以侧面反映反射面颜色的变化（前提是两对红外对管与测量面较接近且等高）[12]。

电磁线寻迹的方案有很多种。比如电磁感应电磁场测量方法、霍尔效应电磁场测量方法等测量方法。他们各自用到的传感器也各式各样，如半导体霍尔传感器、电磁线磁场等传感器。这些测量方法的磁场测量的范围和精度不同，而且各传感器的性价比、尺寸、功耗也不同[13]，在综合考虑这些因素后，本设计采用电磁感应线圈的方案，因为他具有原理简单、价格便宜、体积小、频率响应快、电路实现简单等优点。导线中的电流发生规律性的变化时，导线周边的磁场也会发生相应的变化，进而在线圈中产生了感生电动势。根据法拉第电磁感应定律，电磁线圈磁场传感器的感应电动势 E 与电磁感应线圈的匝数 N 、磁场磁感应强度 $B(t)$ 、电磁线圈横截面积 A 的关系有：

$$E=(NA)\times(\mu_0\mu_r)\frac{dB(t)}{dt}=-\frac{du(t)}{dt} \tag{3.1}$$

需要达到寻迹的目的一个线圈肯定是不够的，因此需要至少一对的线圈才可以帮助智能小车判断小车偏向了电磁线的哪一侧。

摄像头寻迹的方案主要是各种数字摄像头和高度集成的摄像头模块。为了达到快速开发的效果，可以使用 OpenMV 单片机。OpenMV 拥有很多已经封装好的函数，可以较方便地调用他们。可以使用 OpenMV 对车道线进行识别，在两条车道线中间做一条平行于车道线且距离两边的车道线等距的线，这条线就是小车需要寻的轨迹。

由于沙盘已预装了电磁线，因此，首先考虑电磁寻迹的方法。采用此方法能够最先达到最基础的功能，当然也会很实用。而 OpenMV 在其他功能模块中也会用到，也为

OpenMV 预留这个方案，后期考虑可以用摄像头寻迹方案替换掉电磁寻迹方案。

3.2 电磁寻迹方案的硬件设计

采用一对电感来检测电磁线的磁场强度的变化，磁场强度的变化会通过电感两端电压的大小来反映。电感在小车头部的位置分布图如图3.1。

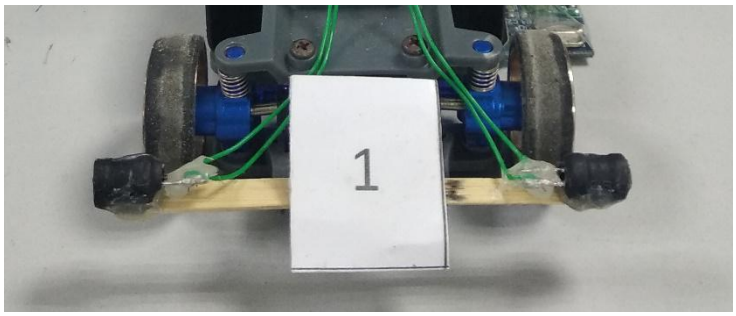


图 3.1 电感在小车头部的位置分布图

但是电感两端的电压信号是非常微弱的，且容易被干扰，直接进行 AD 转化的话误差会非常大。因此需要一些电路来配合优化一下。信号微弱，那么我们就用运算放大器来放大这个信号，波动容易被干扰，那么本文就用滤波电容来稳定它。如图 3.2 所示是在 AD 软件中画得这一部分的原理图。

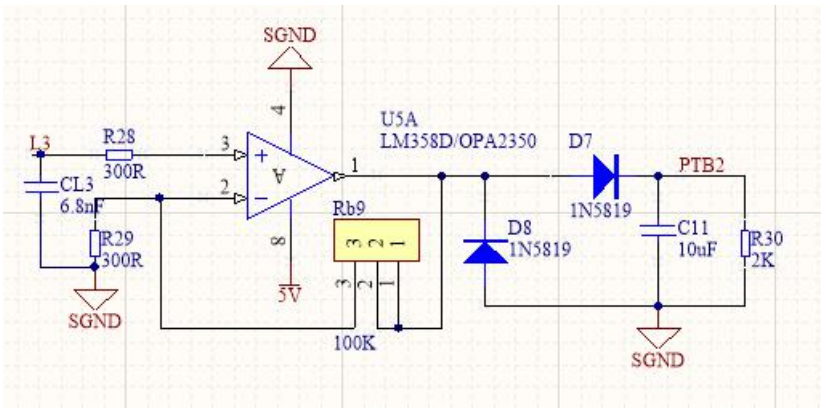


图 3.2 运算放大电路原理图

其中 LM358D 是运算放大器。Rb9 是 100k 的滑动变阻器。1N5819 是一种肖特基二极管。L3 是电感一端的输入端，它与信号地之间的电势差为U_{L3}，根据电路可知这是一个同向比例运算放大器。由“虚短”可知

$$U_{-}=U_{+} \tag{3.2}$$

根据“虚断”可知

$$U_{L3}=U_{+} \tag{3.3}$$

可推得

$$\frac{U_{out}-U_{L3}}{R_{b9}}=\frac{U_{L3}}{R_{29}} \tag{3.4}$$

得

$$U_{out}=(1+\frac{R_{b9}}{R_{29}})U_{L3} \tag{3.5}$$

由此关系可知，这个滑动变阻器 R_{b9} 是用来调整放大倍数的，通过改变滑动变阻器的阻值大小就能改变整个传感模块的灵敏度[14]。运算放大器后面的肖特基二极管是用来防止反向击穿的，以及电容 C_{11} 是用来滤波，减小输出电压的波动的。端口 PTB2 连接至单片机的引脚，将电信号传递给单片机。一共有四组这样的电路，各连接电感的两个引脚。电磁传感器模块引出的端口 PTB2 接至 Arduino Mega 2560 的带 ADC 功能的引脚。将模拟信号转化为数字信号的电路一般称为模数转换器，简称 A/D 或 ADC（Analog to Digital Converter）。A/D 转换的功能是将幅值连续、时间也连续的模拟信号转换为幅值离散、时间也离散的数字信号。模数转换的过程主要包含采样、保持、量化和编码四个过程。模数转化的过程如图 3.3 所示。

图 3.3 模数转化过程

采样就是把随时间连续变化的模拟信号转换为时间离散的模拟信号的过程。但是采样电路将得到的模拟信号转化为数字信号时需要耗费一点时间的[15]，因此为了能够不停地给后面的编码提供稳定值，每次获得的模拟信号需要一段信号保持电路来维持一段时间，这样的过程我们称为保持。在经过采样保持我们得到一个模拟值，而通过以一个最小单位的整数倍来表示这个模拟值，这样的过程我们量化。量化后将数字信号转化成便于信号传输的数字信号的过程我们称为编码。因为实际情况中的传输通道非常复杂，如果仅仅用高电平和低电平来表示数字信号会难适应复杂的通信通道环境，需要采用特别的编码方式来使得信号便于传输。

3.3 电磁寻迹方案的软件设计

小车电磁循迹方案的软件设计程序流程图主要如下图 3.4。

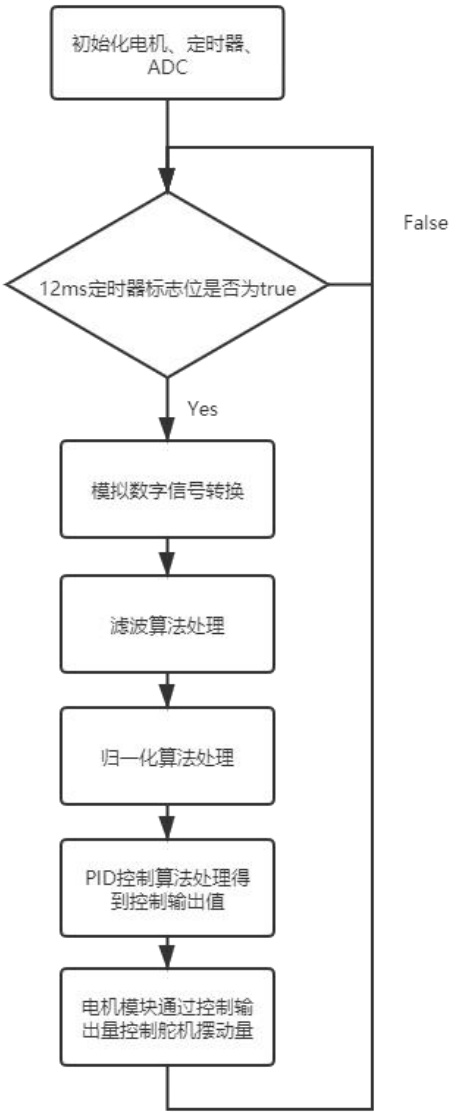


图 3.4 电磁循迹方案的软件设计程序流程图

在 arduino Mega 2560 单片机上编程。loop 循环中一直检测 12ms 定时器的标志位，当标志位为 1 时执行相关操作。其中 Car_Go 是小车行驶状态的标志位。标志位为 0 是代表小车是在行驶的，为 1 时表示小车是停止的，显然是需要在行驶时循迹。

Motor_Test_Flag 是舵机测试状态的标志位，这里不需要关心，它与小车行驶状态标志位是或的关系。

Time_12ms_Flag 是 12ms 定时的标志位。而定时的标志位都来自于定时器终端服务。如上，这是一个 6ms 定时器，每 6ms 会计数一次，通过这个，就可以产生 12ms，24ms，60ms，480ms 定时器。

这里是通过单片机的带模拟信号功能的引脚获取电感引脚的电压值并将其通过单片机的模数转化功能模块转化为数字信号以供单片机芯片进行运算处理。由单片机经过数模信号转换处理得到的数字信号仍然无法直接使用，需要通过滤波算法进行滤波的操作来去除噪点和误差。这里采用的是递推平均滤波法（又被称为滑动平均滤波法）。将连续取得的 N 个采样值看为一个数组，这个数组的数组的长度固定为 N。每当取得一次新的采样值时，就把这个值放入数组中，并把数组中最早放入的值去掉，简而言之就是先进先出的原则。这样这个数组就能表示最新的滤波值。而代码中的 N_i 变量和队列数组就起到了这个先进先出的作用。N_i 通过不断从 0-7 循环不断给队列数组中的变量赋值。因为有四个引脚的电压需要采样，所以需要一个二维数组来分别记录四组采样值，用长度为 4 的一维数组来记录四个实时滤波值。优点是可以抑制规律性的干扰，拥有较高的平滑度；因此它适用于高频率振荡的系统。缺点是灵敏度差，对突然出现的脉冲型干扰的调节作用差；相对浪费随机存储器资源[16]。

```
//**********计算偏差并赋值给舵机**********//
float car_error = 0;          //当前偏差,偏差有正有负
float car_error_last = 0;     //上次偏差
float servo_out = 0;          //float
float real = 0;

void run_error()
{
  //计算偏差
  car_error = (float)((adc_guiyi[0]-adc_guiyi[1])/(adc_guiyi[0]+adc_guiyi[1]));
  car_error = (float)(car_error * 10u);
  //PD
  servo_out = Servo_Kp[Car_Number-1] * car_error + Servo_Kd[Car_Number-1] * (car_error - car_error_last);
  //更新偏差
  car_error_last = car_error;

  //限幅(70-110)
  if (servo_out > 40)  { servo_out = 40; }
  if (servo_out < -40) { servo_out = -40; }
  real = servo_out / 2 + Server_mid_value[Car_Number-1];
  if(real <= 62)  { real = 62; }
  if(real >= 110) { real = 110; }
  myservo.write((char)(real));
}
```

图 3.5 电磁循迹 PID 控制程序

电磁循迹 PID 控制程序的代码如图 3.5。这里用到的算法是 PID 控制算法。PID 控制是近现代发展的最早的控制方法，被广泛用于工业控制中。PID 控制环节分为三个环节：比例环节，积分环节和微分环节。在比例环节中，控制器能够对系统误差进行成比例地反应，当偏差产生的时候，控制器就会通过产生比例控制作用来减小偏差。但是如果仅有比例控制时，会存在静态误差。*Kp* 越大，动态响应的速度就越快，消除偏差的作用效果就会越佳。但是因为现实中的系统基本都是会有惯性的，所以如果比例控制作

用太强就会使系统震荡，不稳定。积分环节中，积分控制器的输出控制值与输入的误差值的积分成正比，误差值的积分越大，控制器的输出值就越大，误差值的积分越小，控制器输出值就越小。简而言之，积分作用的功能就是只要由误差存在，它就会对误差进行不断的积分，使输出值继续增大或者减小，最后使得系统的稳定值无限趋近于给定值，通过这样控制器就能达到消除静态误差的效果。但实际系统是有一定惯性的，所以积分的快慢必须要和系统的惯性相适应。惯性大，积分作用就需要小一点，如果积分作用太大，系统就会产生震荡和积分超调的问题，这就得不偿失了。*Ki*越大，积分作用越强。在微分环节中，控制器的输出控制值和偏差信号值的微分成正比的关系。微分环节表现了输入的偏差信号值的发展趋势，能够在偏差还没有变得太大的时候，给系统插入一个提早的修正值，从而达到减少系统的反应时间的效果。*Kd*越大，微分环节越大。积分环节拥有使系统的响应速度变快，超调量减小，减轻震荡现象的优点，对动态过程有良好的预测、提前调节的作用[17-18]。

在本情景之下，电磁小车的循迹并不需要控制静态误差接近于零，因此对于积分作用的需求不大。而微分作用在对偏差变化量有预测的作用，在例如转弯等情形下有非常好的表现，因此这里的控制器采用的是 PD 控制器。程序中 servo_out 即小车舵机舵量的输出量，car_error 是当前的系统偏差，car_error_last 是上次的系统偏差。由于舵机的摆舵量有限，并且我们也不希望小车太大的转弯，因此对舵量输出进行了一个限幅操作。Server_mid_value[Car_Number-1]这个数组中存了每个小车的舵机舵量中位值，必须确保输出的给定值是正确的，才能得到希望得到的输出值。

3.4 本章小结

本章主要比较分析了三种循迹方案：电磁寻迹寻迹，红外光电寻迹，摄像头寻迹的优缺点。在选择了用电磁循迹的方案后，介绍了本功能方案的硬件设计：通过电感两端的电压来反应磁场的变化，电压通过运算放大器放大后给单片机处理。单片机通过两个电感的电压得知两电感所在位置磁场的强弱。由于电磁线两边的磁场强度应该是对称的，因此两个电感的电压差异就反应了小车偏离电磁线的情况。接着有介绍了本功能方案的软件设计：主要是通过对采集到的 ADC 数据进行滤波归一等处理后，通过 PID 控制算法对小车舵机进行控制来达到循迹的功能。

4 小车程序无线下载模块设计

4.1 小车程序无线下载模块的硬件设计

之前一直使用的是普通的USB-ISP有限下载器，每次烧录程序都需要把小车拿到电脑旁用USB数据线连接上下载，非常的麻烦。因此制作了一个单片机程序无线下载的模块。通过这个模块，只要配对好两个收发模块即可实现无线下载烧录程序的目标。并且原来很难获取小车的实时数据，需要连串口线很是不方便，而通过这个模块可以达到单片机无线数据回传的功能，方便程序调试。

硬件设计主要由STC15W204S、CH340C、E104-BT02组成。STC15W204S这款单片机是STC基于8051内核做的。它有14个I/O引脚、2个定时器/计数器，它的主频最高有35Mhz，它有4K字节的程序存储区，256字节的数据存储区（变量使用），质量上乘的带电可擦可编程只读存储器，ISP/IAP，串口UART，看门狗，CMP等各种丰富的资源。CH340芯片是线路转换的芯片，它能够将USB信号转为串口信号。发射端的工作原理是程序通过Micro USB口传输程序到CH340芯片，CH340芯片将USB信号转化为串口信号，通过E104-BT02传输程序。E104-BT02由单片机STC15W204S控制串口程序的初始化和传输。E104-BT02一款性价比优异的串口转BLE模块，它自带PCB板载天线，工作在2.4GHz频段，模块可以通过串口收发蓝牙数据，数据传输稳定高效，它拥有主从一体的特点，在从机模式下，可以设置为最高57600bps的波特率。并且它还支持AT指令配置模块的功能设置。图4.1是无线下载模块发送端的原理图。图4.2是无线下载模块接收端的原理图。

图4.3是无线下载模块3DPCB图。

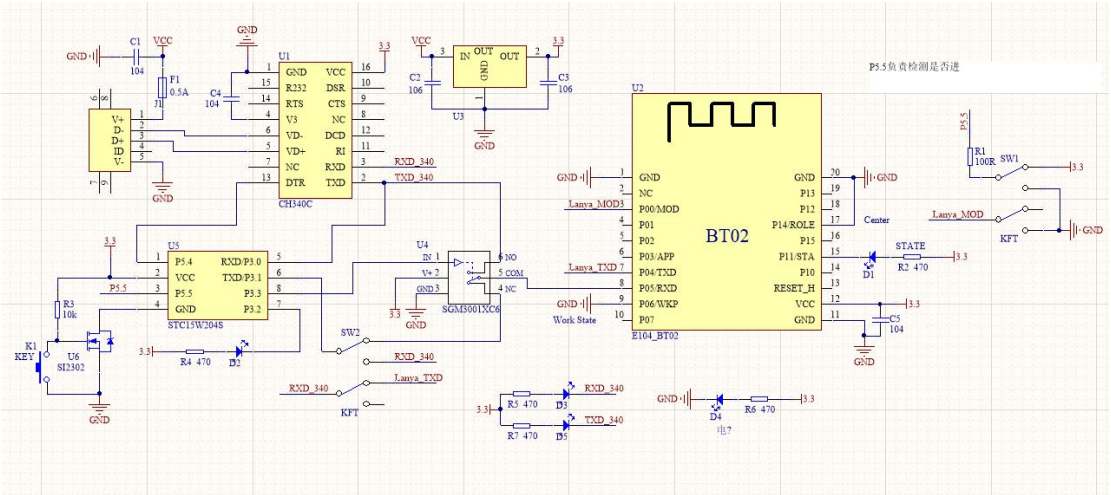


图 4.1 无线下载模块发送端原理图

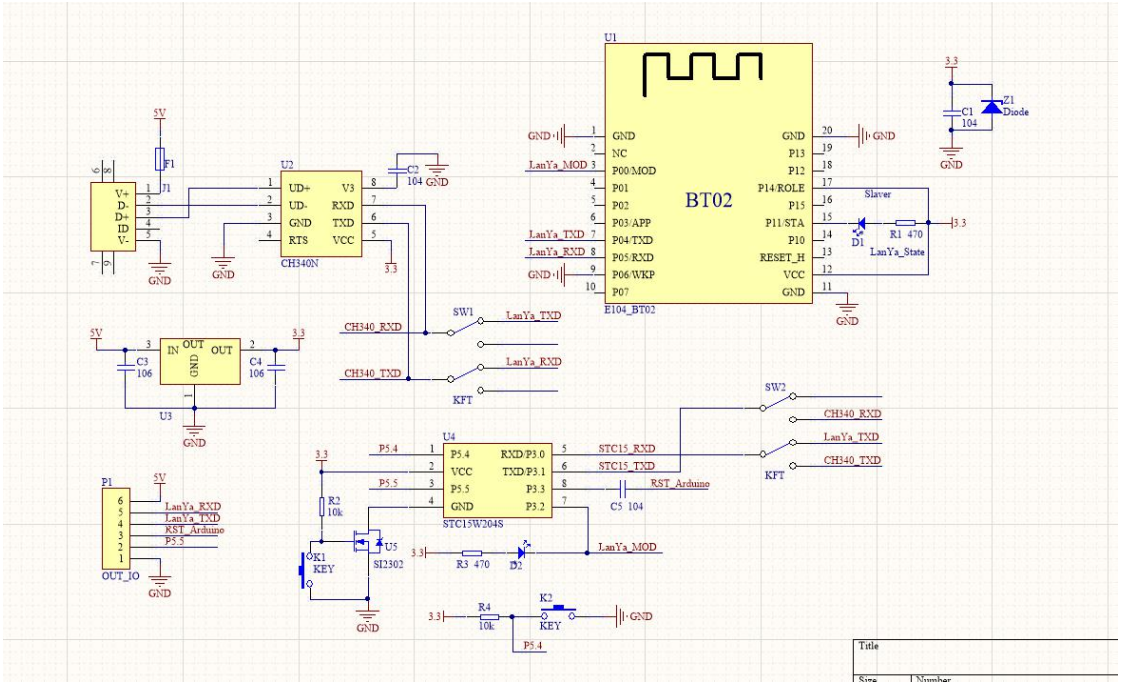


图 4.2 无线下载模块接受端原理图

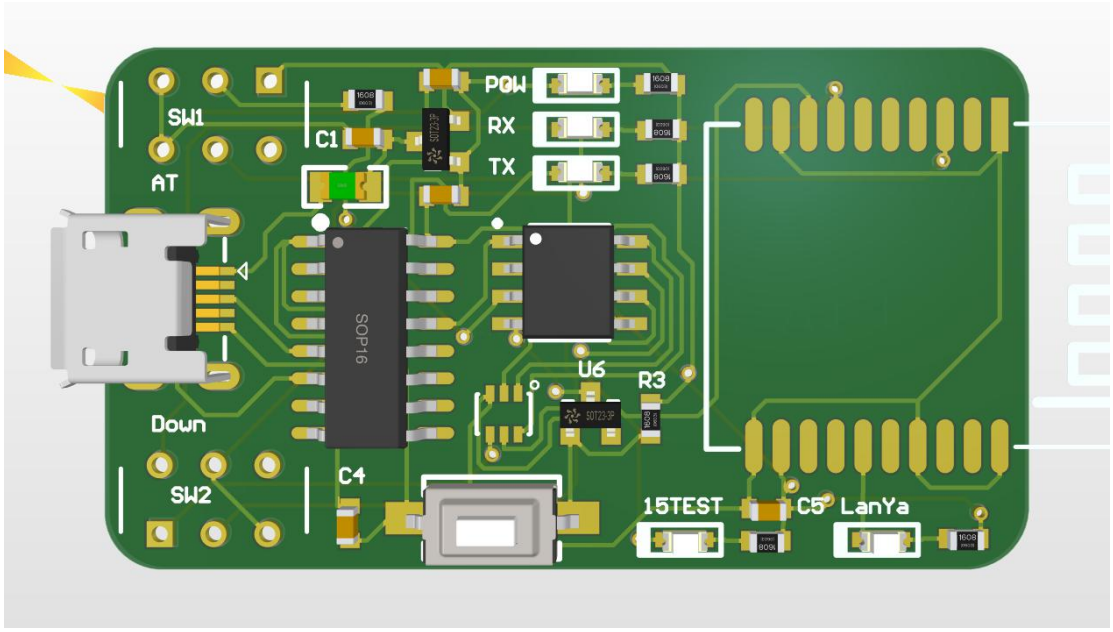


图4.3 无线下载模块3DPCB图

4.2 小车程序无线下载模块的配置以及使用

收发模块蓝牙配置的过程：第一步将 SW1 及 SW2 打到对应位置，使用 MICRO USB 线连接到电脑;第二步，由于新模块波特率默认 19200，主、从机均需修改为 38400。按顺序通过串口发送如下指令复位：<RESET> 恢复出厂设置：<RESTORE> 波特率查询：<COMBAUD> 修改波特率为 38400：<BAUD38400> 这里需要注意：模块重启后配置生效！并不是发送后立即生效。第三步，MAC 地址绑定：原理：从机查询出 MAC 地

址后，将主机绑定到该地址即可自动连接，开启透传。从机 MAC 查询指令：<MAC> 主机 MAC 绑定：<BONDMACXXX> 其中 XXX 为从机地址。解除 MAC 绑定：<DISBOND>。设置 MAC 地址与查询 MAC 地址界面如图 4.4。

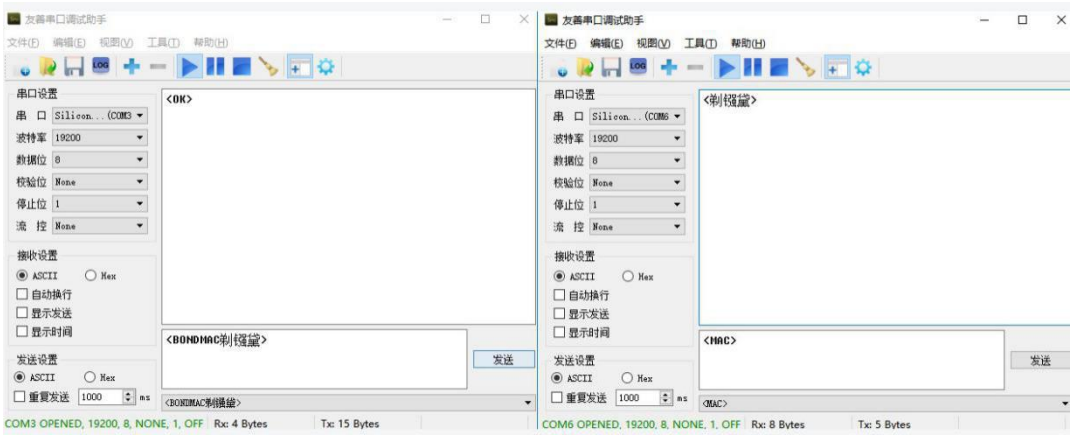


图4.4 设置MAC地址与查询MAC地址界面

表 4.1 发送端开关状态说明

发送端开关、LED 状态与模式对应关系			
模式	SW1	SW2	LED(15TEST)
蓝牙 AT 指令	左	左	无要求
Arduino 下载	右	左	常亮

注：LED 的状态可通过开关 SW1 控制：从右向左拨动一次，闪烁与常亮两状态切换一次。

表 4.2 接收端开关状态说明

接收端开关状态、MOD 小灯与模式对应关系			
模式	SW1	SW2	MOD
蓝牙 AT 指令	左	右	亮
正常下载	左	左	灭
单纯数据透传	左	左	灭
上位机通信	左	右	灭

注：MOD 的状态可通过左边白色按键控制：按动一次，熄灭与常亮两状态切换一次。

之后进行 Arduino 程序下载：首先将下载器接收端与开发板的 TXD、RXD 交叉互联，接收端供电 5V，将接收端的 RST 与板卡的 RESET 相连；开关 SW1 及 SW2 打到对应位置，LED 状态为常亮；接收端用 Micro USB 线连接到电脑，保证已安装 CH340

驱动；

发送端与接收端会自动配对，等待发送、接收端的 LanYa 指示灯均为常亮此时打开 Arduino IDE 软件，选择对应的板卡与端口号即可编译烧录。当然 Arduino 想要实现串口下载必须刷入 Bootloader，购置的裸芯片是没有的，需要后期刷入。Arduino 中选择对应板卡，选择使用的编程器，烧写 Bootloader。（需要等挺久，完成之后可能有红字报错，忽略即可）。Arduino 的 bootloader 其实是社区针对 arduino 板子开发的一小段代码，通过这段代码，即使我们没有外部烧录工具，我们也能够将程序烧录进单片机中。形象的打个比喻，bootloader 类似于我们手机上的 IOS 操作系统，而我们的代码就像是运行在 IOS 操作系统上的各种 app。

但是一开始尝试的时候，发现仍然无法下载代码，到网上查了一些资料发现 Arduino 默认 Mega2560 的 Bootloader 下载波特率为 115200，对于无线下载来说速度较快，会发生数据丢失和错误的情况，因此必须修改 Bootloader 下载速度。Bootloader 的波特率与 IDE 中 board.txt 文件中设置的波特率需要设置一致才可以正常下载。首先需要找到 bootloader 源码：bootloader 的源码都会在这个软件安装根目录\arduino-1.5.2 \hardware \arduino \avr \bootloaders 文件夹下，不同种类的 arduino 板有不同的文件夹。Mega 2560 对应 stk500v2 文件夹。

进入 skt500v2 文件夹，删除其中的.hex 文件，然后用文本编辑器打开.c 文件，修改 #define BAUDRATE 后的波特率，修改完成保存退出。

最后，通过命令窗口进入当前文件夹，输入 make mega2560 的指令，回车后就会产生新的 hex 文件。到此，整个下载的配置操作全部完成。

4.3 本章小结

本章首先讲了设计这个小车无限程序下载模块的原因：许多小车放置在平台上，每个小车通过连线烧录程序会比较费时又不便。接着介绍了小车程序无线下载模块的硬件设计：主要由单片机STC15W204S、USB转串口芯片CH340C、蓝牙串口透传模块E104-BT02组成。分两块板，一块是发射端，接受USB传来的电脑的程序。一块是接收端，接受蓝牙模块传来的数据，再将程序数据烧录到小车上的Arduino控制板中。最后介绍了无线下载模块配置的相关操作，还分析解决了一个中途碰到的问题：下载波特率太快导致程序下载失败。

5 通信网络模块分析与设计

5.1 通信网络模块设计方案选择

由于交通调度半实物仿真平台的场地是在室内，因此主要考虑的就是两种通信方式了：蓝牙通信方式和WiFi通信方式。在考虑稳定性和信息传输速度后，决定采用WiFi串口的传输方式。选用的模块是淘宝网购买的ESP-8266-12E无线串口模块。它支持完整的TCP/IP协议栈，能够将WiFi功能嵌入其他系统。更方便的是，它支持用AT指令进行网络配置。

ESP-8266-12E的外观及引脚图如图5.1。

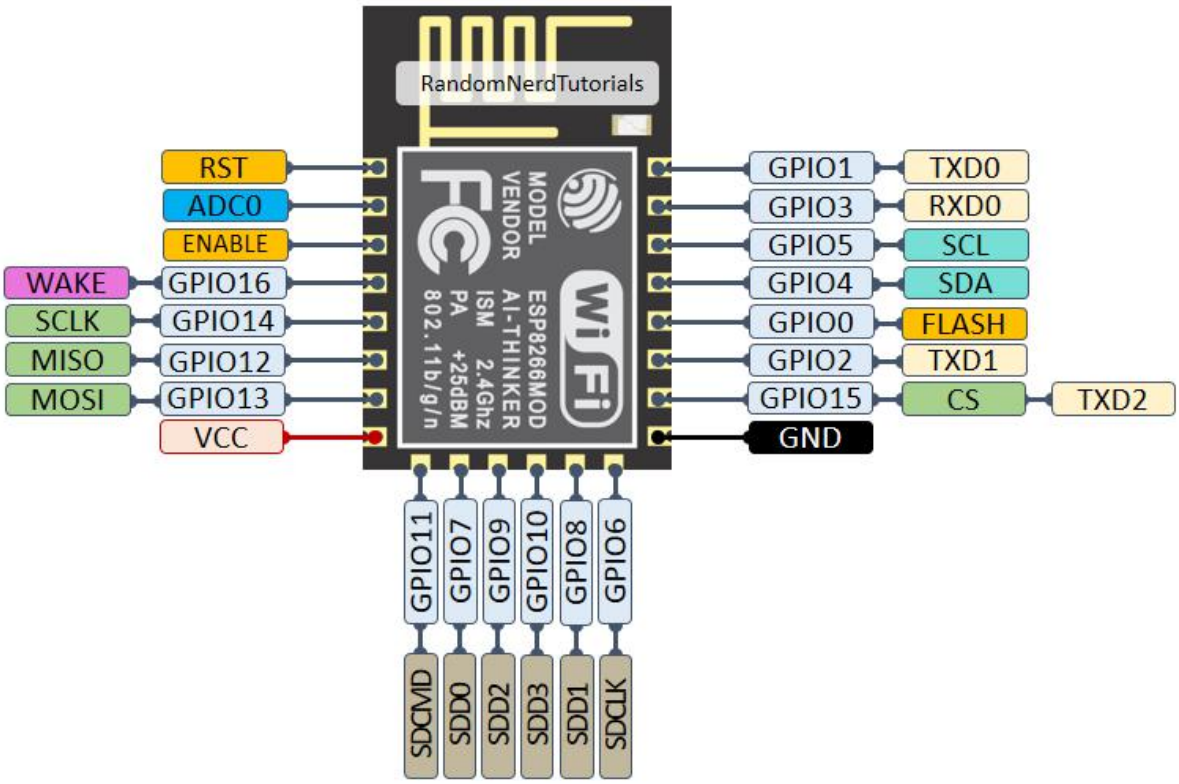


图5.1 ESP-8266-12E的外观及引脚图

5.2 通信网络模块以及系统相关功能设计

5.2.1 通信网络基础构建设计

数据接受程序流程图如图5.2。

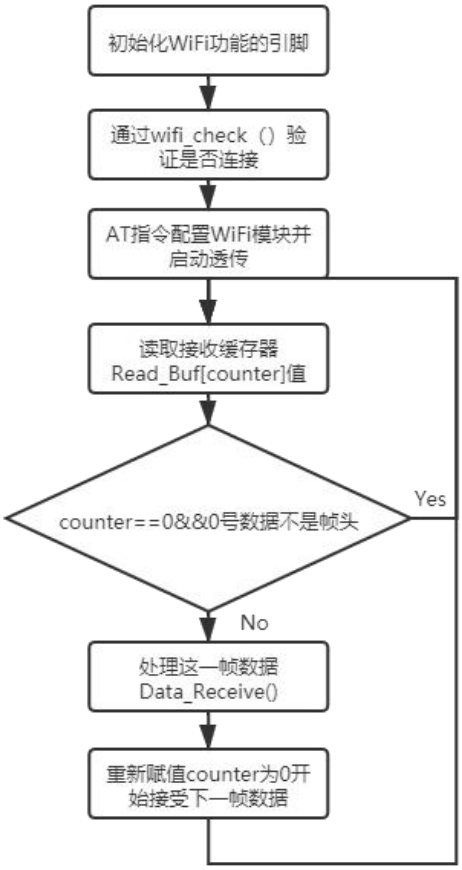


图 5.2 数据接受程序流程图

首先在setup里初始化Wifi：主要步骤有设置波特率为115200：

Serial2.begin(115200);

波特率是对符号传输速率的一种度量单位，它表示每秒钟传送的码元符号的个数，要想能够传输正确的信息，就得匹配上波特率，不然就会出现乱码或者丢包的情况。

然后是通过匹配IP地址以及端口来连接上位机以及WiFi：

Serial2.println("AT+CIPSTART=\"TCP\",\"192.168.1.190\",5555");

通过wifi_check（）函数来验证是否成功连接上WiFi。

然后给模块发送AT命令设置透传模式以及开启透传：

单片机串口接受上位机（或者说Wifi）发来的消息是有协议的。有了通信的协议，就能够高效、稳定地接受消息。在这里定义一帧地帧头是0xFE，帧尾0xEF

使用counter来计数，并且通过检测帧头是否是0xFE、帧尾自动重新赋值counter来确保得到的每一帧数据都是完整的，不完整的会抛掉并且再努力使下一帧数据完整。

在数据的预处理函数Data_Receive()中更清楚地解释了通信的协议以及一帧数据的

结构。数据预处理函数程序图如图5.3。

```
switch(Read_Buf[3])
{
  case 0x01:  //设置智能汽车参数
      Car_Speed_Receive = Read_Buf[6];
      Run_Step = 0;  //执行到第几个节点计数归零,执行新指令
      for(char i=0;i<((Data_Length-1)/2);i++)  { Node[i]=int(Read_Buf[2*i+7]<<8|Read_Buf[2*i+8]); }
      Node_Num = (Data_Length-1)/2; //收到的节点数量统计
      Node_Data_Dispose();  //处理收到的节点数据
      Serial2.write(0xFE); Serial2.write(0x00); Serial2.write(Car_Position_1); Serial2.write(0x01);  //控制码
      Serial2.write(0x00); Serial2.write(0x01); Serial2.write(0x01); Serial2.write(0xEF);  //确认数据已接收
      break;
  case 0x02:  //运行参数设置
      Car_Go = Read_Buf[6]; //启停控制
      Serial2.write(0xFE); Serial2.write(0x00); Serial2.write(Car_Position_1); Serial2.write(0x02);  //控制码
      Serial2.write(0x00); Serial2.write(0x01); Serial2.write(0x01); Serial2.write(0xEF);  //确认数据已接收
      break;
  case 0x03:  //状态查询
      for(char i=0;i<Data_Length;i++)  { Status_Inquiry[i]=Read_Buf[i+6];}
      Serial2.write(0xFE); Serial2.write(0x00); Serial2.write(Car_Position_1); Serial2.write(0x03);  //控制码
      Serial2.write(0x04); Serial2.write(0x00); Serial2.write(0x00); //指令码
      Serial2.write(Car_Speed); for(char i=0;i<(Data_Length-1)/2;i++) { Serial2.write(char(Node[i])); Serial2.write(char(Node[i])>>8); }  //发送设置参数
      Serial2.write(0x01); Serial2.write(Car_Speed); Serial2.write(0x02); Serial2.write(char(Card_Data)); Serial2.write(char(Card_Data>>8)); Serial2.write(0xEF);
      break;
  case 0x04:  //中心对状态确认
      Register_Confirm = Read_Buf[6];
      break;
```

图 5.3 数据预处理函数程序图

Read_Buf{1}和Read_Buf{2}是用来存小车的地址码的，而小车的地址码是一个十六位的数，串口的传输是以字节为单位传输的，因此需要两个字节的数字组成这个地址码，其中高位存在Read_Buf{1}，低位存在Read_Buf{2}。同理数据的长度也是同理，其高位存在Read_Buf{4}，低位存在Read_Buf{5}。由程序可见，Read_Buf{3}就是指示帧类型的变量。小车通过读取Read_Buf{3}的值来知道这一帧数据是用来干什么的。帧数据的类别主要可以分为设置小车参数、运行参数设置、状态查询、注册状态确认。并且这样的设计为未来的功能预留了位置，使拓展新的需求变得规范、便利。具体各帧数据类别的内容在后面阐述。

5.2.2 整个通信模块对于整个平台功能的作用体现：

（1）车辆向上位机注册功能

在Car_Confirm()函数里，小车上电后就会向上位机发送注册信息，其中的Car_Position_1就是小车的编号，我们对实验室的小车从1开始都进行了编号。此外，函数里面还有一个3秒的定时器，每隔三秒会确认是否有注册成功，注册成功的标志位是Register_Confirm，他的值在一帧数据的第七位即Read_Buf{6}（当Read_Buf{3}==0x04的时候）。

case 0x04: //注册状态确认

Register_Confirm = Read_Buf[6];

break;

（2）车辆向上位机返回速度以及位置信息的功能

通过Position_Return() 函数实现小车实时信息向上位机发送的功能。除去帧头帧尾等主要发送了小车的速度信息和小车的位置信息。

Serial2.write(Car_Speed);

Serial2.write(char(Card_Data));

其中Car_Speed代表小车的速度，Card_Data代表识别到的射频卡的十进制编号。

（3）车辆接受参数设置以及路径规划信息获取功能

Read_Buf{6}里包含了由上位机发来的设置小车速度的信息，Node{i}里面即存了路径规划的信息，一个结点Node也是十六位的，因此也需要两个字节来传输高低位。

5.3 本章小结

本章主要首先比较了蓝牙通信模块和 WiFi 通信模块的优缺点。在选择 WiFi 通信模块为本设计的硬件方案后。介绍了代码中 WiFi 通信模块的使用以及通信模块中我们设计的通信传输中帧的格式。以及整个通信模块在整个平台中起到的三个主要功能作用。

6 位置信息获取以及路径规划功能设计

6.1 位置信息获取以及路径规划功能硬件设计

MFCR522是高度继承的非接触式读写卡芯片。模块与arduino主控板之间的通信方式是SPI（串行外设接口）通信。通过MFCR522IC读卡模块，我们可以读取接近的射频卡的信息。

6.2 位置信息获取以及路径规划功能软件设计

首先通过mfrc522.PICC_IsNewCardPresent()函数并判断读卡器上是否由新卡，没有就会返回主程序，不占用资源。通过mfrc522.PICC_ReadCardSerial()判断是否已经读取NUID。在读到新卡和NUID后就会进入try_key函数。

Try_key函数中，主要做了上报位置的操作：

if(Card_Data != Last_Card_Data)　{ Position_Return(); }　//卡片数据更新则上报位置

通过Position_Return函数，上位机就得到了这辆小车的信息和位置。

接下来看路径规划方面，在前面通信模块部分我们已经得到了存储路径规划信心的节点数组Node[]。

节点Node的每一个元素都是十二位的数据，首先提取出它的前四位，它代表的是方向信息。并将方向信息先存到Go_Direction[]数组中。

接下来介绍小车如何进行路径规划指令的执行：如图6.1是小车路径规划的代码。

```
/********利用上位机数据进行路线规划********/
void Path_Planning()
{
    if(Go_Direction[Run_Step] == 4)  //倒车指令
    {
       Run_Step = Run_Step+1;
       Node_Num = Node_Num-1;
       Car_Go = 2;    //倒车入库专用
    }
    if(Card_Data == ((Node[Run_Step]<<4)>>4) && Node_Num > 0)
    {
       if(Go_Direction[Run_Step] == 0) { Car_Go = 0; }     //发车
       if(Go_Direction[Run_Step] == 1)
       {
           if( Card_Data == 0x34D ) { Turn(-16.5,60); }
           if( Card_Data != 0x34D ) { Turn(-15,60);}  //左转
       }
       if(Go_Direction[Run_Step] == 2)    //右转
       {
           Turn(19,60);
           //if( Card_Data == 0X347 ) { Turn(19,60); } //出库特殊处理
           //if( Card_Data != 0X347 ) { Turn(18,60); }
       }
       if(Go_Direction[Run_Step] == 3) { Car_Go = 1; }     //停车
       Run_Step = Run_Step+1;
       Node_Num = Node_Num-1;
    }
    if(Node_Num == 0)   { Run_Step = 0; }
}
```

图 6.1 小车路径规划的代码图

Go_Direction[]中已经存了方向信息，并且他就是Node数组元素中的前四位。通过(Node[Run_Step]<<4)>>4）我们就去掉了Node数组每一个元素的高四位（即方向信息）这样就只剩下后面的12位，即发送过来的指定IC卡编号。如果这个小车在指定的IC卡处，就会根据GO_Direction数组中存的方向信息进行操作，主要是前进左转右转等。小车达到每一个IC卡，都会知道应该做什么操作，这就达到了路径规划的功能。

6.3 本章小结

本章主要介绍了位置信息获取以及路径规划功能的硬件设计，选用了MFCR522IC读卡模块作为核心硬件模块。然后介绍了位置信息获取以及路径规划功能的软件设计，通过软件将读卡模块读取到的IC卡编号上传到上位机，那么上位机就知道了小车的位置信息。接下来介绍了路径规划功能的实现：小车得到一帧串口数据，这其中包含了指定的卡片编号和对应的动作，经过一系列数据处理，小车根据读到的IC卡信息，来进行指定的动作，这样就实现了路径规划的功能。

7 交通信号灯识别功能设计

7.1 交通信号灯识别功能的硬件设计

交通信号灯识别功能的硬件设计主要是采用OpenMV摄像头。通过OpenMV摄像头与主控单片机Arduino的硬件连接来传输控制信号。OpenMV是一款低成本、低功耗、小巧的电路板，通过它可以帮助我们轻松地完成与机器视觉相关的应用开发。它采用的是MicroPython进行编程，Python拥有高级数据结构以及丰富的库，能够赋予我们能力解决复杂的图像处理问题。但他本质仍然是一款单片机，处理器处理图像信息后会通过IO口或者通信模块将控制信号输出。它搭载的是STM32H743II ARM Cortex M7 高性能处理器。并且它还拥有可拆卸的摄像头模块系统。我们的硬件设计中采用的是OV5640感光元件来处理2592*1944（5MP）图像，在QVGA（320*240）以及以下分辨率时运行大部分算法都可以达到二十五到五十的帧率。OV5640拥有体积小、工作电压低等优点。我们根据需求也可以更换不同的镜头到它的镜头底座上[19]。

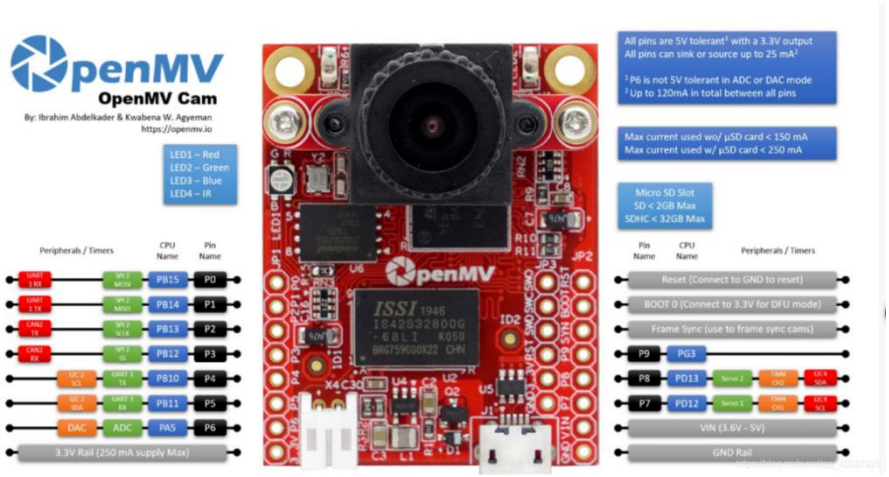


图 7.1 OpenMV模块的引脚说明图

另一个便利之处是，我们可以通过USB线连接OpenMV观察摄像头的识别效果，同时可以调试程序来改进识别程序的图像识别结果。图7.1是OpenMV模块的引脚说明图。

7.2 交通信号灯识别功能的软件设计

使用OpenMV运行microPython程序来达到图像处理识别的功能。如图7.2是小车交通信号灯识别的程序流程图。

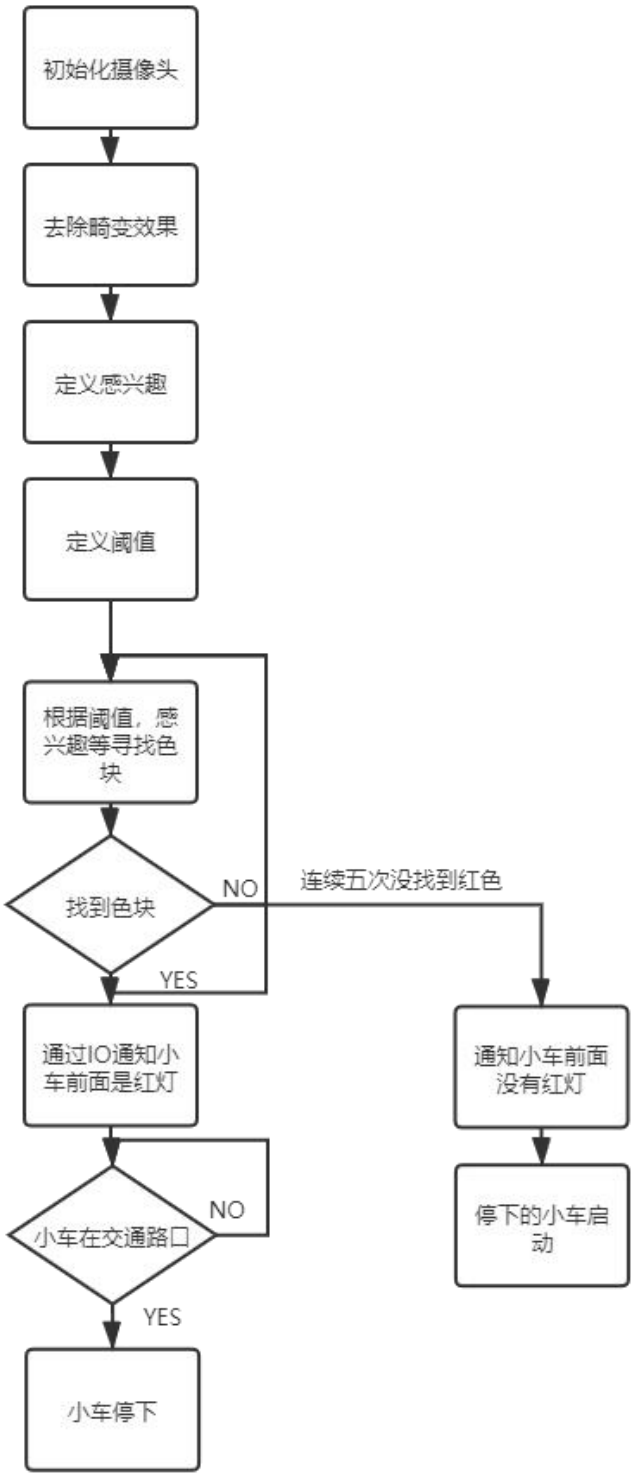


图 7.2 小车交通信号灯识别的程序流程图

首先调用 *img = sensor.snapshot().lens_corr(strength = 1.8, zoom = 1.0)*

来获取一帧图片并对图像去畸变。其中strength是一个浮点数，该值确定了对图像进行去鱼眼效果的程度。zoom是在对图像进行缩放的数值。

再调用

blobs=img.find_blobs([red_threshold],x_stride=1,y_stride=1,roi=roi1,merge=True,pixels_threshold=2)

寻找色块函数来找红色色块。thresholds为颜色阈值，是一个元组，需要用括号［］括起来，里面的值为LAB的参数，LAB颜色空间中，L亮度；A的正数代表红色，负端代表绿色；B的正数代表黄色，负端代表兰色。不像RGB和CMYK色彩空间，LAB颜色被设计来接近人类视觉。因此L分量可以调整亮度对，修改a和b分量的输出色阶来做精确的颜色平衡。这里我们设置红色的阈值red_threshold=(93，100，-22，3，5，61)，这个阈值是通过OpenMV IDE 中的机器视觉栏中的阈值编辑器手动调试得到的。roi设置颜色识别的视野区域，roi是一个元组， roi = (x, y, w, h)，代表从左上顶点(x,y)开始的宽为w高为h的矩形区域。这里为了减少不必要的噪点和误差，我们只取小车通过路口时，交通信号灯在摄像头图片中显示的区域，设置roi1=(62,52,31,26)。x_stride是查找的色块的x方向上最小宽度的像素，y_stride是查找的色块的y方向上最小宽度的像素。Merge：合并，如果设置为True，那么合并所有重叠的blob为一个。pixels_threshold是像素个数阈值，如果色块像素数量小于这个值，会被过滤掉[20]。图7.3，7.4是OpenMV阈值编辑器。

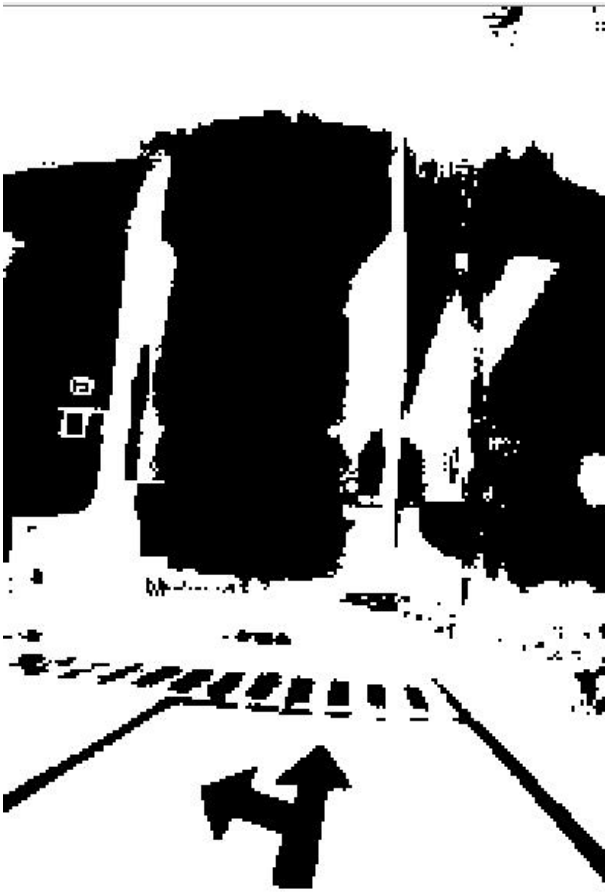


图 7.3 OpenMV阈值编辑器（一）

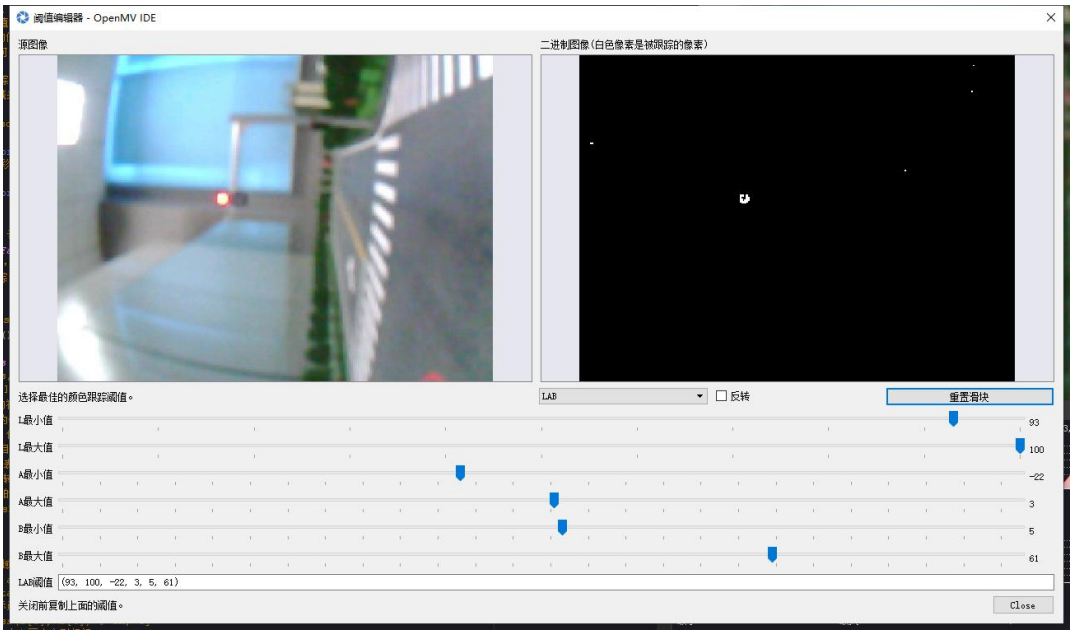


图 7.4 OpenMV阈值编辑器（二）

find_blobs这个函数返回一个数组，{0}代表识别到的目标颜色区域左上顶点的x坐标，{1}代表左上顶点y坐标，{2}代表目标区域的宽，{3}代表目标区域的高，{4}代表

目标区域像素点的个数，{5}代表目标区域的中心点x坐标，{6}代表目标区域中心点y坐标，些值接口非常丰富，只是为了能让我们在IDE中看到图片的识别效果来方便我们的调试。主要用到的是设置p0口为高电平让小车知道我们碰到红灯了。变量n的作用是当连续5张图片都没找到红色说明没有红色的交通信号灯了，就需要把p0口的高电平状态改变成低电平。

在arduino 单片机方面，需要一个引脚来从OpenMV获得信息。

```
int Red_Stop_IO = 32;        //红灯IO PC5
pinMode(Red_Stop_IO, INPUT);
```

初始化arduino的PC5口为使用的IO口并且将它设置成输入模式。

Get_Light_State里面存了八个路口的射频卡编号，小车只会在这八张卡的位置做是否要因为红灯停的决策。单片机通过digitalRead（32）来读取连接OpenMV单片机的那个引脚的高低电平。Red_Light就是小车因红灯停车状态的标志位，1是停车。当小车在这八张卡的前后两张卡的范围内检测到电平信号为低，那么就取消小车需要因红灯停车的状态。

7.3 本章小结

本章开始介绍了交通信号灯识别功能的硬件设计：OpenMV单片机的硬件配置介绍。然后介绍了交通信号灯识别功能的软件设计：阈值设置操作、感兴趣区域设置操作、寻找色块函数、小车如何获得来自OpenMV单片机的信息以及交通信号灯识别程序的运行情景。

8 避障功能设计

8.1 避障功能设计的方案选择

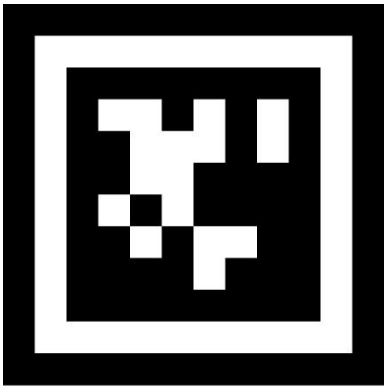
现在主流的智能小车避障方案有三种：超声波避障、红外避障、摄像头避障。

超声波避障一般采用超声波测距模块实现自主避障。该模块可以实现非接触距离测量，测量的精度可以达到厘米级别。避障模块包括超生波发射器，接收器与控制电路，包括VCC、GND、TRIG、ECHO 4个端口。超声波测距的工作原理是通过内部产生一定频率的信号，经发射器发射后接收器检测是否有信号被物体反射回来，以此来判断前方是否有障碍物及与障碍物的距离。

红外避障采用红外传感器来检测前方障碍物情况，避障功能是通过发光二极管通电后发射出红外光，碰到障碍物后，红外光被遮住或有部分红外光被反射回来，确认前方是否有障碍的方式是：红外光电二极管负责接收被反射回来的红外光，将其转化为相应的电信号，并与之前的信号做对比。

超声波模块避障对于近的物体避障效果不佳，可能会影响小车的避障效果，而红外避障模块对远处的物体避障效果不佳，可能给小车预留的反应时间过短来不及反应。本文采用的摄像头避障主要是因为已经采用摄像头模块来进行交通信号灯的识别，如果避障功能再添加一个硬件模块，就会加大整体硬件系统的复杂度。而且基于在我们仿真的沙盘上，小车是巡电磁线行驶的，没有其他障碍物，唯一的障碍物是其他小车，因此只需要和其他的小车保持车距即可。并且如果代码调试得好，远近的避障效果都能达到不错的水平。本文的摄像头避障功能是基于密歇根大学的AprilTag算法设计的。AprilTag是一个视觉基准系统，可用于各种任务，包括增强现实、机器人和摄像机校准。目标可以从普通打印机中创建，AprilTag检测软件可以计算标签相对于相机的精确三维位置、方向和身份。AprilTag库是用C实现的，没有外部依赖关系。它的设计可以很容易地包含在其他应用程序中，也可以移植到嵌入式设备上。简而言之，我们可以通过将这个Apriltag贴到目标上，就可以在OpenMV上识别出这个标签的3D位置，偏转角度和ID。

所用的AprilTag如下，TAG36H11是它的家族，它的家族中还有其他tag成员，不同的成员的tag都不一样，因此实现了编号的功能。图8.1是其中一个AprilTag码。



TAG36H11 - 0

图 8.1 AprilTag码

8.3 避障功能的软件设计

小车避障功能的程序流程图如图8-2所示。

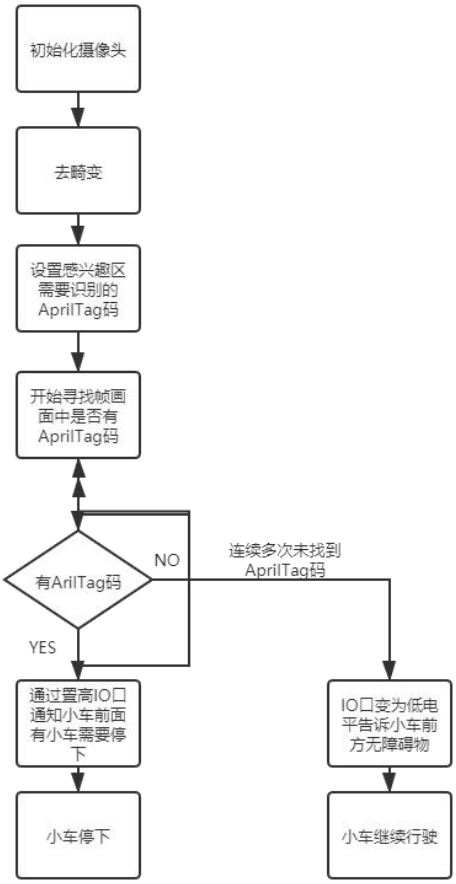


图 8.2 小车避障功能的程序流程图

OpenMV方面，首先调用find_apriltags这个函数找到apriltag码，并在电脑上显示的

实时图片上标注出来方便调试。tag.x_translation()函数可以返回和识别到的apriltag码的距离。这样我们就可以设定一个阈值，当距离小于一个给定量时，将另一条与arduino相连的IO口置高电平来通知arduino需要因为避障而停车。如果在之后的帧图像中没有找到apriltag码或者距离在阈值之外那么相应的IO口也会恢复为低电平。当然这里也为之后预留了改进方案，根据距离的大小改变车速。距离与车速成正比，距离小于一定程度的时候车速为0，这样就不会产生突然停下的突兀感。小车识别AprilTag码的效果图如图8.3。

图 8.3 小车避障的效果图

Arduino 方面，Barrier_Stop 是小车需要因为避障而停车状态的标志位，它通过单片机读取相对应的引脚就会改变至相应的状态。由代码可见 Car_Go、Barrier_Stop 、Red_Light 只需要一个标志位满足等于 1 就会让小车通过改变电机的 pwm 输出值而停止。

8.3 本章小结

本章先讨论了三种避障方案（超声波避障、红外避障、摄像头避障）的优劣势后结合本仿真平台的实际情况，选择了摄像头避障的方案。通过 OpenMV，基于 AprilTag

技术对 AprilTag 标签进行识别，通过识别到的距离信息来控制小车是否需要停下。

结　　论

通过本次毕业设计，大体实现了初级阶段的智能交通半实物仿真实验平台的建设。展示结果显示小车队伍可以借助电磁线自行从停车场出发，在道路上平稳行驶。在 OpenMV 单片机的帮助下，智能小车车队能够在交通路口对交通信号灯进行判断，也能够实现与前方的车辆保持安全的车距。通过 Wifi 模块，小车能够接收上位机发送来的路径规划指引来达到倒车入库调整速度的目的。在整个毕业设计过程中，发现了很多问题，也不断对硬件软件以及方案进行了改进。为实验室之后性能更完善、优异的智能交通半实物仿真平台建设打下了一点基础。

致　　谢

首先要感谢郭戈老师给我这个机会能够加入到实验室中，并且为我们提供资源来推进这个项目。然后要感谢李啸天学长的帮助，和学长一起为了平台展示日以继夜的调试改进的日子学到了很多，也留下了宝贵且美好的记忆。也感谢实验室的高振宇老师以及学长学姐们提供的方方面面的帮助，特别是高振宇老师不论在这个项目中还是生活的一点点滴中都不遗余力地帮助过我。也特别感谢计算机学院的老师帮我们平台做的 PC 上位机的软件。本科毕业就在眼前，但是前路漫漫，道阻且长，之后要踏入异国他乡，希望自己也能够不忘初心，更加努力，打磨自己，走向卓越。

参考文献

[1] 赵娜,袁家斌,徐晗.智能交通系统综述[J].计算机科学,2014,41(11):7-11+45.

[2] 王旭辉.国内外城市智能交通系统的发展概况及启示[J].山东工业技术,2018(15):200-201.

[3] 张瑜亮.交通大数据在智能高速公路中的应用探讨[J].中国交通信息化,2021(S1):238-239.

[4] 柳妍.智能交通系统在交通运输管理中的应用[J].时代汽车,2021(10):184-185.

[5] 骆珍仪.智能交通仿真平台的设计与实现[J].电脑知识与技术,2017,13(33):270-272.

[6] 花瑞. 交叉口交通信号控制半实物仿真平台的设计[D].武汉理工大学,2008.

[7] 马雪晴,殷传志.一种基于 STM32 的智能交通信号灯设计的研究[J].南方农机,2021,52(09):175-176+183.

[8] https://www.forum8.co.jp/chinese/uc-win/Road-robocar-cn.html

[9] 陆正辰. 基于多缩微车的智能交通系统仿真平台研究[D].上海交通大学,2013.

[10] 赵津,张博,张庆余,赵鹏超.基于城市先进道路交通系统的智能交通仿真平台设计[J].汽车工业研究,2019(01):25-28.

[11] 苏致远,马育林,李建市,周晶晶.基于多缩微车的智能交通半实物仿真平台[J].兵工自动化,2016,35(08):5-8.

[12] 张楚翘.自动寻迹小车的设计与实现[J].科技视界,2021(04):4-7.

[13] 张克明,毕春光.电磁寻迹车的设计与实现[J].信息技术与信息化,2014(09):168-171.

[14] 詹立春.基于 NE555 的幅频可调发生器的设计[J].电子制作,2016(11):74-75.

[15] 张旭. 重磁数据采样技术分析与试验[D].中国地质大学(北京),2020.

[16] 10 种软件滤波方法.https://blog.csdn.net/FUBIN0000/article/details/75697222

[17] 刘魏晋,胡琛,唐洋,李树杰,陈俊杰.基于 PID 算法实现三级储罐液位控制[J].南方农机,2021,52(09):114-115.

[18] Xiaojun Zhou. A Rolling PID Control Approach and Its Applications[A]. 中国自动化学会控制理论专业委员会（Technical Committee on Control Theory, Chinese Association of Automation）、中国自动化学会（Chinese Association of Automation）、中国系统工程学会（Systems Engineering Society of China).第三十八届中国控制会

议论文集（2）[C].中国自动化学会控制理论专业委员会（Technical Committee on Control Theory, Chinese Association of Automation）、中国自动化学会（Chinese Association of Automation）、中国系统工程学会（Systems Engineering Society of China):中国自动化学会控制理论专业委员会,2019:5.

[19] 黄洋,王瑞,潘新奇,蓝升传.基于 ARM 嵌入式的关于图像处理的交通信号灯识别[J].电子制作,2019(15):82-84.

[20] 施敏虎,栗云鹏,庄曙东,符正帆,王齐鑫.基于 OpenMV 的智能搬运车型机器人的设计[J].机械工程师,2020(02):20-22+25.

附　　录

附录 A 英文原文

Intelligent Transportation Systems

The ever-increasing need for mobility has over-flooded major cities with vehicles, resulting in growing traffic congestions, accompanied by unpredicted emergencies and accidents. These facts reveal important inefficiencies related to transportation, which call for the development of systems for more efficient and safer mobility. A way to pursue this is to apply the recent findings in the area of communications to the field of transportation management. This can be facilitated by exploiting cognitive networking principles, i.e., by developing transportation management mechanisms with learning capabilities to enable the a priori perception of potential dangers and accordingly amendment of vehicle's behavior. This article proposes functionality that is capable of exploiting the intelligence accumulated through the exchange of information among numerous vehicles that lie in a certain vicinity. This leads to decisions that improve the quality of transportation in terms of reduction of traffic congestions, accident risks, and emergency situations.

The increasing need for mobility has brought about significant changes in transportation infrastructures. European cities are thus more and more overcrowded with vehicles, facing unpleasant everyday phenomena such as growing traffic congestions, as well as unpredicted emergencies and accidents [1]. Inefficiencies cause enormous losses of time, decrease in the level of safety for both vehicles and pedestrians, high pollution, degradation of quality of life, and huge waste of nonrenewable fossil energy [1], [2]. These inefficiencies bring up the necessity for developing systems for more efficient and safer mobility. In response to the above, traffic assessment and management has been established as a key service that should be offered in the area of transportation by Information and Communication Technologies [3]–[4][5]. In this respect, several innovative and cost-effective mobile services and applications for traffic networks are under investigation, emerging as the cornerstone of the so-called intelligent transportation systems (ITS) [6], [7]. Despite the establishment of ITS, there are ways to maximize transportation efficiency and safety:

- The traffic conditions that should be handled by vehicles may frequently change in a sudden or recurring manner. So, on one hand, traffic needs to be assessed in real time. On the other hand, traffic patterns resulting from a learning process could add accuracy to the messages communicated to the drivers.
- Legacy traffic assessment and management systems are mainly centralized. This means that, in principle, they are complex and unsuitable for adapting, in short-time scales, to context changes.
- Currently, the collection of context information, the solution of optimization problems, and the application of reconfiguration decisions is an off-line process applied in medium- or long-time scales.
- Intelligence embedded in vehicles is still at a very low level, and there is no assessment in the vehicle of the overall safety status that would rely on a correlation of the global traffic condition and the vehicle and driver behaviors.

A direction for obtaining adaptability to new traffic contexts, facilitating cooperation and also addressing complexity, is to apply, as much as possible, distributed and yet cooperative solutions, at the same time, exploiting experience and knowledge (placed locally in the vehicle and also globally in the network). This can be realized by introducing cognitive systems [5] for the

management of vehicles. By definition, cognitive systems can retain knowledge from past interactions with their environment, transform this knowledge to experience, and plan their future actions accordingly. This can improve the performance (e.g., resolve congestion /emergency situation faster) and also the reliability of any decisions taken.

The scope of this article is to introduce novel functionality for providing knowledge to vehicles, thus jointly managing traffic and safety. This will be achieved through the design of the proposed functionality, which, at a high level, will comprise 1) sensor networks formed by vehicles of a certain vicinity that exchange traffic-related information, 2) cognitive management functionality placed inside the vehicles for inferring knowledge and experience, and 3) cognitive management functionality in the overall transportation infrastructure. The goal of the aforementioned three main components shall be to issue directives to the drivers and the overall transportation infrastructure valuable in context handling.

The rest of the article is structured as follows. The “Motivation” section presents the motivation for this work, providing an overview of the wireless world and focusing on cognitive systems and wireless sensor networks (WSNs). The “Functionality Description” section presents the proposed management functionality in detail, i.e., the components, as well as an indicative information flow among them. Finally, the last section delivers some concluding remarks and outlooks for future work.

Motivation

This section provides an overview of the current wireless world, focusing on cognitive wireless networks and systems. Then, it describes the basic principles of WSNs and explains how WSNs can cooperate with cognitive systems in building a functionality that will act as an intelligent transportation system that can enhance traffic and safety management in vehicles and the overall transportation infrastructures.

Wireless Landscape

The wireless world has been migrating toward the so-called beyond the third-generation (B3G) era.” From a technological perspective, in the B3G era, legacy (conventional) access network technologies, called radio access technology (RAT) standards, coexist and cooperate with currently emerging as well as completely new standards. In this respect, today'
s wireless world comprises numerous RATs of diverse nature, which can be classified as follows:

- wireless wide-area networking technologies, which include, among others, second- and third-generation mobile communications [8], [9], the more recently introduced worldwide interoperability for microwave access [11], as well as broadcasting technologies such as digital video broadcasting and digital audio broadcasting (DAB) [12]

- wireless networking technologies of a shorter range, which include, among others, wireless local- and personal-area networks (WLANs/WPANs) [10], as well as wireless ad hoc networks and WSNs [13], [14].

Obviously, it is not possible to stand still. Even 4G systems are on the way. In the near future, even higher bit rates will be supported. However, more significant than the bit rates are the capabilities of future networks that include full integration of Internet Protocol, even smaller cells, self-planning dynamic topologies, flexible use of the spectrum, and utilization of precise user location. As to trends in services, it will become more and more important to deliver the right information at the right time and to the right place.

However, as can be seen day to day, novel communication systems become more and more complex. Complexity is usually derived 1) from the heterogeneous network and terminal infrastructure that needs to be tackled every time and 2) to the continuously increasing level of complexity of services and applications that arise from the ever-increasing user expectations for dependable, reliable, and secure services. The deployment of high-complexity systems can be facilitated through several concepts, one of which is the reconfigurability concept, often seen as an evolution of software-defined radio (SDR) [15]. Reconfigurability provides the technologies that are essential for terminals and network elements to dynamically (online) select and operate with those RATs that are considered as most appropriate for tackling the specific conditions encountered in a certain region and time zone. RATs are SDR based, i.e., they can be installed and uninstalled only through the appropriate activation and deactivation of the respective software components.

Moving one step further, complexity can be fought through the design of communication infrastructures on the premises of cognitive networking principles [5]. In general, a cognitive system is capable of retaining knowledge from past interactions with the external environment and decides upon its future behavior based on this knowledge, other goals, and also policies, so as to adapt to optimize its performance [5]. It is anticipated that cognitive systems can facilitate the design, development, and integration of novel services and applications. An area of applications where cognitive systems could find prosper ground is transportation. Indicatively, a cognitive system placed inside a vehicle might seem like the one shown on Figure 1.

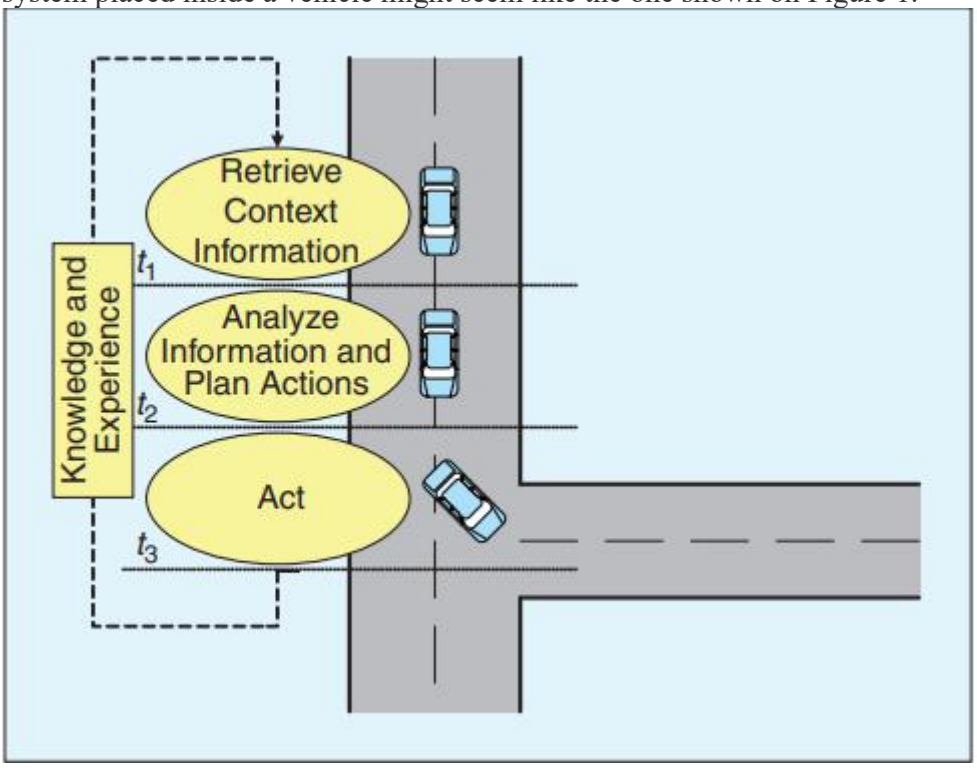


Figure 1 - Operation of a cognitive system.

Complexity can be fought through the design of communication infrastructures on the premises of cognitive networking principles.
As shown in the figure, the operation of a cognitive system placed inside a vehicle can be reflected on a feedback look. The system at time t1 retrieves context information, potentially on traffic, velocity of neighboring vehicles, etc. Through the analysis of this information (at time t2), while taking into consideration its own preferences, goals, and policies, the system (at time t3) decides on its actions (e.g., issue a directive toward the driver to change the vehicle's direction).

The output of the system is stored on a knowledge database, which might simply be a matrix for future reference. This means that the system keeps track of its actions so as to learn from their implications to facilitate future decisions. This is repeated in a machine-learning process [16] that leads to cognition.

In general, the reconfigurable and cognitive systems move toward the most promising directions and technologies in the sense of removing any potential limitations that derive from business perspectives and providing the means to implement the vision of true end-to-end connectivity.

Wireless Sensor Networks

As mentioned earlier, WSNs [13], [14] form part of the B3G wireless world. Such a network may comprise hundreds of nodes that operate on the grounds of small batteries, and thus, its viability may depend on the resources consumption on behalf of its nodes. WSNs may exist in areas where certain measurements such as temperature, pressure, humidity, and velocity need to take place. Each time a sensor (node) receives a trigger, it forwards the relevant information to the whole network (WSN). Other sensors (nodes) receive this information and keep on forwarding it up to a point where one or more SINK nodes, i.e., nodes that have less energy limitations, larger processing power, and thus undertake to gather any significant information (also further process it to an external network) (see Figure 2).

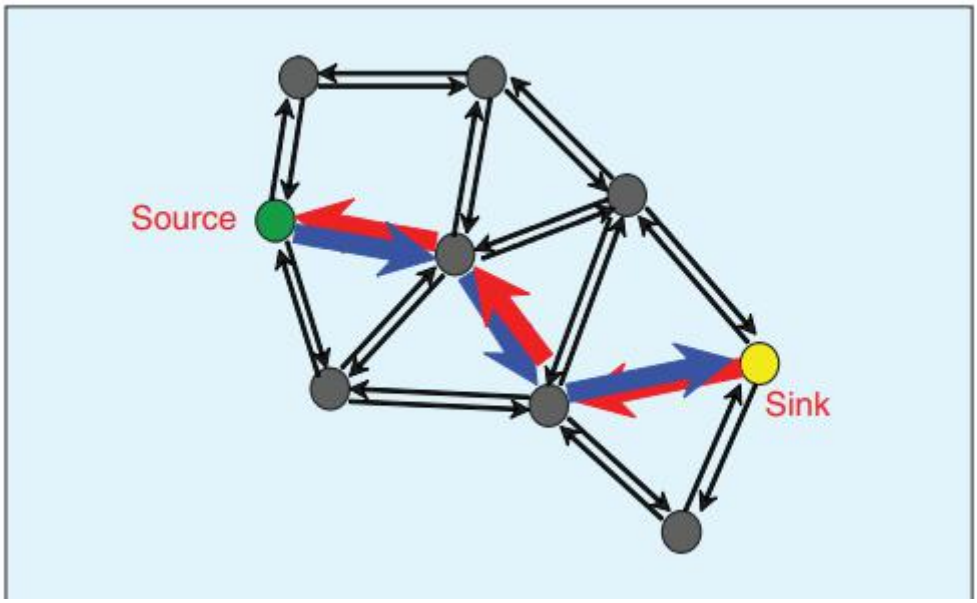


Figure 2 - Information transfer in a WSN.

The way information is forwarded inside the WSN may vary, since there are many options for packet routing. What is more, nodes may be (intentionally) moving, while some nodes may cease to operate because of energy problems. The above render information transfers a tough problem.

In general, WSNs have many exciting applications. In this respect, the following section shows how WSNs can form part of intelligent transportation systems in conjunction with cognitive systems.

Overview of Proposed Functionality

The objective of this article is to propose the design of management functionality that comprises cognitive communication systems to exploit the (collective) intelligence accumulated through the exchange of information among numerous vehicles that lie in a certain vicinity.

In general, this will be achieved through networks of sensors dynamically formed by vehicles that fall at a certain geographical range and capable of allowing communication among the different network nodes (vehicles). This will lead to important information exchange among the network vehicles. Such information may not only improve the quality of transportation in terms of reduction of traffic congestions, but may also be important in automatically reducing accident risks and emergency situations. The functionality is shown at a high level on Figure 3 and presented in detail in the next section.

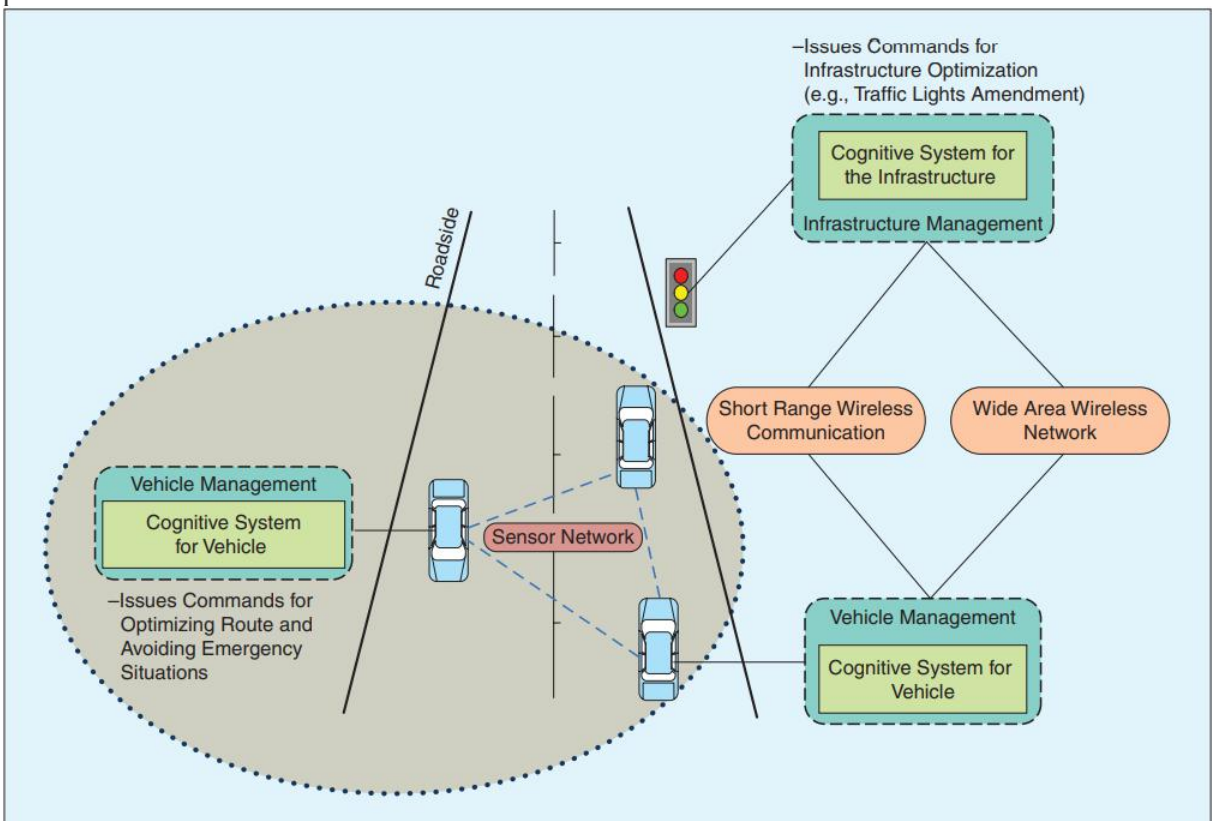


Figure 3 - High-level view of functionality.

Functionality Description

Requirements

The most significant goal of intelligent management functionality for vehicles, as well as for the whole transportation infrastructure, is to improve the levels of efficiency and safety of mobility. In this respect, there are several lateral requirements that need to be tackled before the design and development process:

- awareness of contextual situations: to identify the current context and help the vehicle adapt to it dynamically, securely, and fast and, as such, provide the maximum possible levels of quality
- personalization: not only to support various classes of vehicles/drivers but also to provide solutions tailored to the individual driver profile
- support of pervasive computing: to enable the existence and operation of sensors, ad hoc networking entities, and also local-area networks in all application areas
- always-best connectivity: for providing seamless network access for enabling the functionality's operation in heterogeneous environments
- collaboration with alternate RATs: for enabling the seamless operation of the functionality
- scalability: the ability to provide solutions at various levels to be able to act either in a collaborative or in an autonomic manner depending on the specific needs.

Vehicle sensors are capable of enabling the ad hoc formation of networks among neighboring vehicles to allow the communication and information exchange.
A solution that derives from the analysis of the aforementioned requirements, for the cognitive functionality for the management of vehicles, is presented in the subsequent section.

Architecture and Description of Components
The architecture of the proposed functionality is shown in Figure 4. As shown in the figure, the functionality comprises the following complementary components.

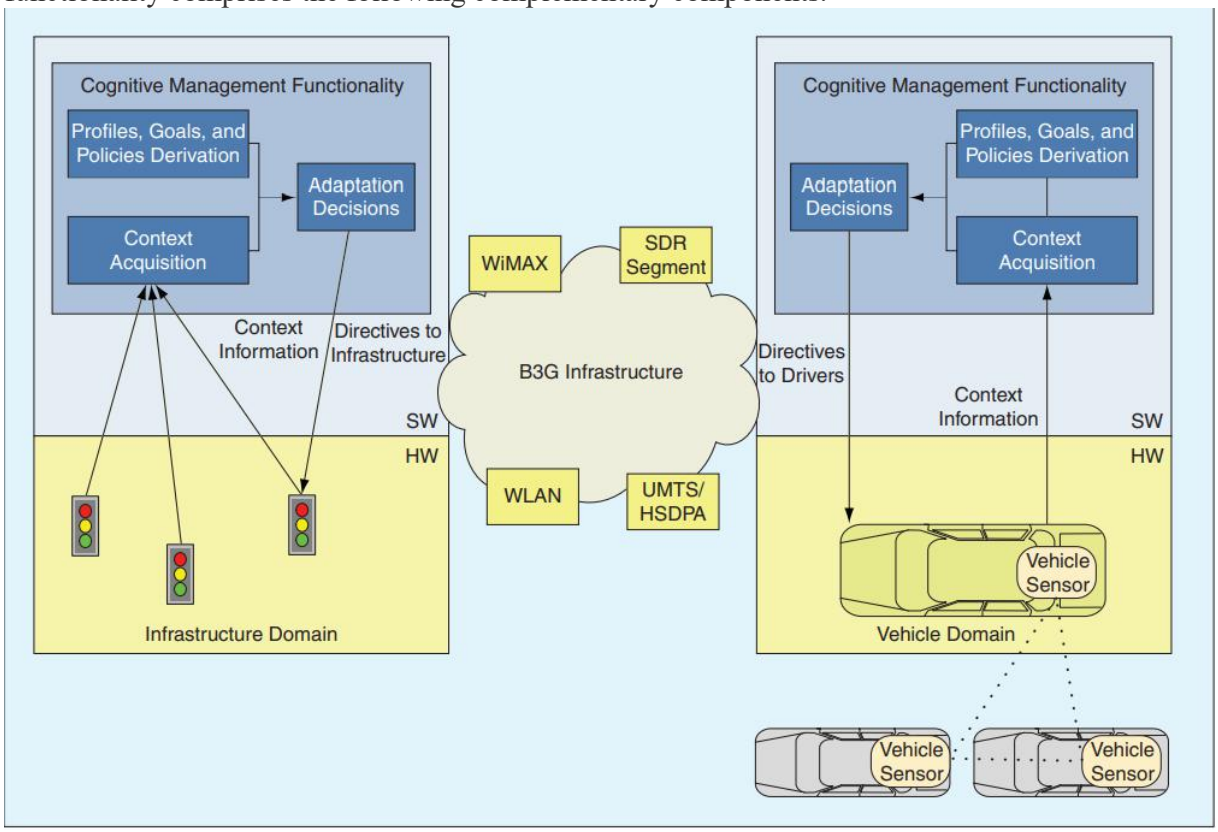


Figure 4 - Architecture of proposed functionality.

Vehicle Sensors and WSNs
Vehicle sensors are capable of enabling the ad hoc formation of networks (WSNs) among neighboring vehicles to allow the communication and information exchange. In general, a vehicle may comprise several sensors. The information exchanged among them can be classified into high- and low-level data.

High-Level Data
Goals and policies aim at maximizing the performance, safety, reliability, and stability of the decisions taken from an end-to-end perspective.
This type of information includes knowledge on the congestion level, alert regarding potential emergencies, characterization of driver's behavior, characterization of vehicle's overall condition (good, normal, and bad) and cruising behavior, information on the road condition (e.g., slippery), information on neighboring vehicles and their cruising behavior, and general knowledge on location (mountain road, city road, etc.).

Low-Level Data
This type of information includes information on the vehicles such as their accurate positions (distance among them), velocities and directions, capabilities braking distances, and accelerations.

Furthermore, information on the drivers is also included, such as the driver's profiles, driving habits, capabilities, and preferences (policies). The resulting information can be used in the inference of a driver's current tasks as well as driver modeling; i.e., common driving related tasks are recognized by the interpretation of sensor measurements of vehicle–driver interactions (e.g., operating of the foot pedals, gear changes, pressing buttons on the instrument panel, or looking into mirrors), as well as state information about the vehicle (e.g., location, fuel levels, and engine status). This process is performed unobtrusively without disturbing the driver in his natural behavior. The focus is on recognizing short-term tasks that a driver may pursue.

In general, the sensors are required to decide on how to process in-vehicle data, which aggregated data are to be sent, how often, etc. Sensor measurements are processed in a hierarchical manner with specialized reasoning techniques, which yield information about the vehicle–driver interactions at various abstraction levels.

Vehicle Cognitive Management Functionality

Input

The vehicle cognitive management functionality (V-CMF) input includes contextual information acquired from the sensors and the WSNs, regarding the status of the vehicle (such as its velocity, direction, neighboring vehicles' positions, directions, and velocities) and road-side information (such as road condition, congestion levels and potential emergencies), as well as traffic lights and road-signs conditions. Moreover, the input includes information on the driver's profiles. To do so, a predefined set of driver states is inferred from interpreted driver-monitoring data (this information is also retrieved from the sensors). Moreover, plan-recognition techniques are explored to derive driver state and behavior. This means that sequences of interactions between the driver and the vehicle, the raw signals about driver's physical condition (eye blink frequency, eyelid opening, head movement, profile, operating the foot pedals, pressing buttons on the instrument panel, and steering wheel activity), and vehicle-state information are also acquired in the form of a facial driver recognition, which allows for the detection of differences between changing driving styles. Finally, driver's goals, priorities, and policies are also included. Goals and policies aim at maximizing the performance, safety, reliability, and stability of the decisions taken from an end-to-end perspective.

Output

The V-CMF results in issuing commands (directives) toward the driver so as to adapt the vehicle's road behavior and tackle any emergency situations through emergency braking or vehicle direction correction (again based on perception and reasoning). Moreover, congestion can be avoided through the reconsideration of the vehicle's advisable route.

Decision Making

Several approaches can be envisaged for the decision-making process. In general, the V-CMF uses appropriate intelligent algorithms that exploit the input in terms of optimizing an objective function [17], [18] that refers to certain aspects of the vehicle's behavior (overall delay, mean velocity, etc.).

Knowledge and Experience

The information acquired is processed and appropriately interpreted so as to infer knowledge and experience. The knowledge model aims at capturing various aspects such as the driver's state (attentiveness and fatigue) and behavior, as well as the overall vehicle environment. Moreover, information on certain contextual situations (recurrent or emergencies) and the way they have been confronted is retained to serve for future decisions.

Infrastructure Cognitive Management Functionality

Input

The infrastructure cognitive management functionality (I-CMF) acquires input from the WSNs regarding the condition of elements or segments of the transportation infrastructure (traffic lights, road signs, road conditions, congestion levels, and overall load in telecommunications network) so as to be aware of the current context. Moreover, the input includes information on the vehicle's profiles, as well as goals and policies dictated by the transportation authorities.

Output

The I-CMF is targeted at deciding on the proper configuration of small elements or larger segments of the transportation infrastructure, i.e., traffic lights and road signs.

Decision Making

Several optimization algorithms are envisaged for the I-CMF. Those algorithms are targeted at achieving optimal performance, safety, reliability, and stability from the end-to-end perspective. Additionally, cost factors are also addressed through the minimization of the overall load in the telecommunication network.

Knowledge and Experience

Decisions need to be enhanced with learning capabilities so as to accelerate and improve the efficiency of the necessary adaptation (reconfiguration) actions. In particular, algorithms are enriched with knowledge features through the incorporation of basic learning techniques, such as pattern matching and context recognition, that help a system compare current contextual situations with past ones and identify already applied solutions that could be put into effect. However, faster, more effective, and more stable leaning strategies can also be adopted.

Cost factors are also addressed through the minimization of the overall load in the telecommunication network.

It should be noted that distributed context acquisition and decision making at various degrees of distribution are enabled by the proposed architecture, i.e., either autonomously from the V-CMF or in a collaborative manner from the I-CMF. In general, the aforementioned components cooperate with each other so as to generate knowledge from various sources and result in useful directives toward the vehicles and the transportation infrastructure. The way this is realized is described in the next section.

Indicative Information Flow

This section aims at exemplifying the operation of the proposed functionality. In this respect, Figure 5 depicts a scenario for showcasing the exchange of information among the functionality's components.

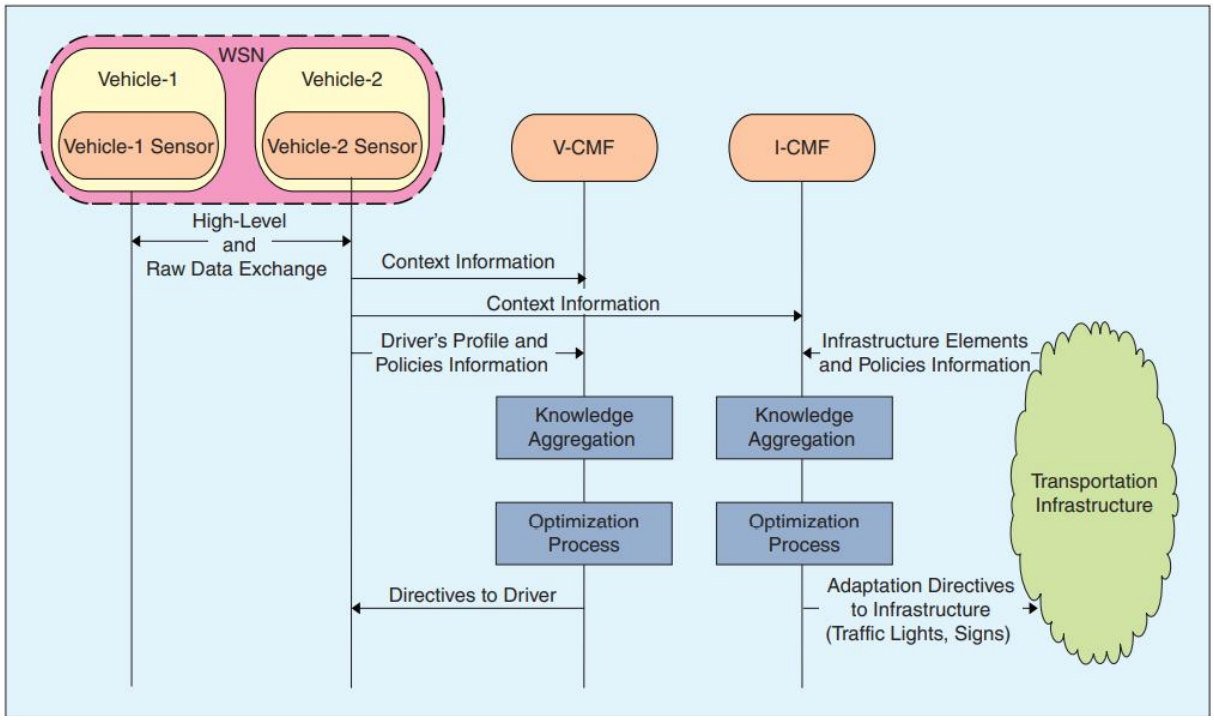


Figure 5 - Functionality components and indicative information flow.

The scenario is twofold in the sense that it comprises 1) a part that is tackled by the V-CMF and 2) a part that is tackled by the I-CMF. In both parts, the initial trigger is supposed to originate in the WSNs. The WSNs exchange information among their nodes (sensors).

In the first part of the scenario, information that is derived from the WSNs, i.e., context information along with vehicle's and driver's profiles, goals, and policies, is transferred to the V-CMF. The V-CMF gathers this information and compares it with data already retained in its knowledge database, to compare the current context (and the relevant decisions taken) with past ones and thus use past knowledge and experience before taking any decisions. Then, the V-CMF runs an optimization process, as described earlier, so as to take decisions and issue any necessary commands (directives) to the driver, valuable in context handling.

The most significant goal of intelligent management functionality for vehicles is to improve the levels of efficiency and safety of mobility.
In the second part of the scenario, again WSNs send information on the current contextual situation to the I-CMF. A main difference here is that the I-CMF also gathers information from the transportation infrastructure regarding the condition of the elements (traffic lights, road signs, etc.) and larger segments (road sides) of the infrastructure. What is more, the I-CMF gathers information on profiles, goals, and policies from the transportation authorities, which constitute rules that need to be taken into consideration by the I-CMF. The I-CMF then compares the information gathered with past contexts confronted and the associate decisions taken. Finally, the I-CMF runs its own optimization process and results in decisions regarding the configurations of the infrastructure to optimize certain criteria such as resources consumption, resolution of an incident, and minimization of cost.

Conclusions
The latest trends in wireless communications envisage technologies that are capable of exploiting knowledge from previous interactions with their environment in planning their future actions. These advances have promoted the seamless integration of information of various types from

transportation networks to benefit drivers and provide several innovative applications. In this respect, the article has provided an overview of a novel ITS based on cognitive systems and WSNs. The functionality proposed is capable of providing valuable directives to drivers, as well as to elements of the transportation infrastructure (e.g., traffic lights), catering for a more efficient and safer mobility.

This work could evolve through a detailed analysis of the interfaces among the proposed components of the functionality to thoroughly describe the information exchanged among them. Moreover, part of our future activities will be dedicated to the utilization of specific optimization algorithms in the context of the cognitive management functionality components to showcase their effectiveness.

附录 B 中文译文

智能交通系统

日益增长的交通需求使主要城市的车辆泛滥，导致交通拥堵加剧，并伴随着无法预料的紧急情况和事故。这些事实揭示了与交通有关的重要低效问题，这就要求开发更有效和更安全的交通系统。实现这一目标的一种方法是将通信领域的最新发现应用于运输管理领域。这可以通过利用认知网络原理来实现，即通过开发具有学习能力的交通管理机制来实现对潜在危险的先验感知，并相应地修改车辆的行为。本文提出的功能，是能够利用积累的情报，通过在众多车辆之间的信息交换，在一定的附近。这导致了在减少交通拥堵、事故风险和紧急情况方面提高运输质量的决策。

对流动性日益增长的需求给交通基础设施带来了重大变化。因此，欧洲城市的车辆越来越拥挤，面临着日益严重的交通拥堵等令人不快的日常现象，以及无法预料的紧急情况和事故[1]。效率低下会造成巨大的时间损失，降低车辆和行人的安全水平，造成高污染，降低生活质量，以及不可再生化石能源的巨大浪费[1]，[2]。这些低效率使得开发更有效和更安全的交通系统成为必要。针对上述情况，交通评估和管理已被确立为信息和通信技术在交通领域应提供的一项关键服务[3]-[4][5]。在这方面，一些创新的、具有成本效益的交通网络移动服务和应用正在研究中，它们是所谓智能交通系统（ITS）的基石[6]、[7]。尽管建立了ITS，但仍有一些方法可以最大限度地提高运输效率和安全性：

- 应由车辆处理的交通状况可能会突然或反复频繁地发生变化。因此，一方面，交通需要实时评估。另一方面，学习过程中产生的交通模式可以提高向驾驶员传达的信息的准确性。
- 传统的流量评估和管理系统主要是集中的。这意味着，原则上，它们是复杂的，不适合在短期内适应环境的变化。
- 目前，环境信息的收集、优化问题的解决以及重构决策的应用是一个离线过程，在中长期范围内都有应用。
- 嵌入在车辆中的智能仍然处于非常低的水平，并且没有对车辆的整体安全状态进行评估，这将依赖于全球交通状况以及车辆和驾驶员行为的相关性。

获得对新交通环境的适应性、促进合作和解决复杂性的一个方向是，尽可能多地应用分布式但合作的解决方案，同时利用经验和知识（在车辆中本地放置，也在网络中全局放置）。这可以通过引入用于车辆管理的认知系统[5]来实现。根据定义，认知系统可以保留过去与环境交互的知识，将这些知识转化为经验，并相应地规划未来的行动。这可以提高性能（例如，更快地解决拥塞/紧急情况）以及任何决策的可靠性。

本文的范围是介绍为车辆提供知识的新功能，从而共同管理交通和安全。这将通过拟定功能的设计来实现，该功能在较高水平上将包括 1）由某个附近的车辆组成的传感器网络，该传感器网络交换交通相关信息，2）放置在车辆内部用于推断知识和经验的认知管理功能，3）整体交通基础设施的认知管理功能。上述三个主要组成部分的目标应是向驾驶员和整个交通基础设施发出指令，这在环境处理中是有价值的。

本文其余部分的结构如下。“动机”部分介绍了这项工作的动机，概述了无线世界，重点介绍了认知系统和无线传感器网络（WSNs）。“功能描述”部分详细介绍了提议的管理功能，即组件，以及组件之间的指示性信息流。最后，在结语部分对今后的工作进行了总结和展望。

动机

本节概述当前无线世界，重点介绍认知无线网络和系统。然后，描述了无线传感器网络的基本原理，并解释了无线传感器网络如何与认知系统合作，构建一个智能交通系统，以增强车辆和整个交通基础设施的交通和安全管理。

无线景观

无线世界正在向所谓的超越第三代（B3G）时代迁移。“从技术角度来看，在 B3G 时代，传统的接入网技术，即无线接入技术（RAT）标准，与当前新兴的和全新的标准共存和合作。在这方面，当今的无线世界包括许多不同性质的老鼠，可分为以下几类：

- 无线广域网技术，其中包括第二代和第三代移动通信[8]，[9]，最近推出的全球微波接入互操作性[11]，以及数字视频广播和数字音频广播（DAB）等广播技术[12]
- 较短范围的无线网络技术，其中包括无线局域网和个人区域网（wlan/wpan）[10]，以及无线自组织网络和 wsn[13]，[14]。

显然，这不可能静止不动。就连 4G 系统也即将推出。在不久的将来，将支持更高的比特率。然而，比比特率更重要的是未来网络的能力，包括完全集成互联网协议、更小的小区、自规划动态拓扑、灵活使用频谱和利用精确的用户位置。就服务业的发展趋势而言，在适当的时间和地点提供适当的信息将变得越来越重要。

然而，正如人们每天所看到的，新型的通信系统变得越来越复杂。复杂性通常是从异构网络和终端基础设施衍生出来的，这些基础设施需要每次处理；2）由于用户对可靠、可靠和安全服务的期望不断增加，服务和应用程序的复杂性不断增加。通过若干概念可以便利部署高复杂系统，其中一个是可重构概念，通常被视为软件无线电（SDR）的一种演变[15]。可重构性提供了终端和网络元素动态（在线）选择和操作那些被认为最适合处理特定区域和时区中遇到的特定条件的老鼠所必需的技术。老鼠是基于 SDR 的，即只有通过适当的激活和停用相应的软件组件才能安装和卸载它们。

进一步推进，可以通过在认知网络原则的前提下设计通信基础设施来克服复杂性[5]。一般来说，认知系统能够从过去与外部环境的交互中保留知识，并基于这一知识、其他目标和政策来决定其未来行为，以便适应其性能的优化[5]。预期认知系统可以促进新服务和应用的设计、开发和集成。认知系统可以找到繁荣的应用领域是交通。从某种意义上说，放置在车内的认知系统可能与图 1 所示的系统类似。

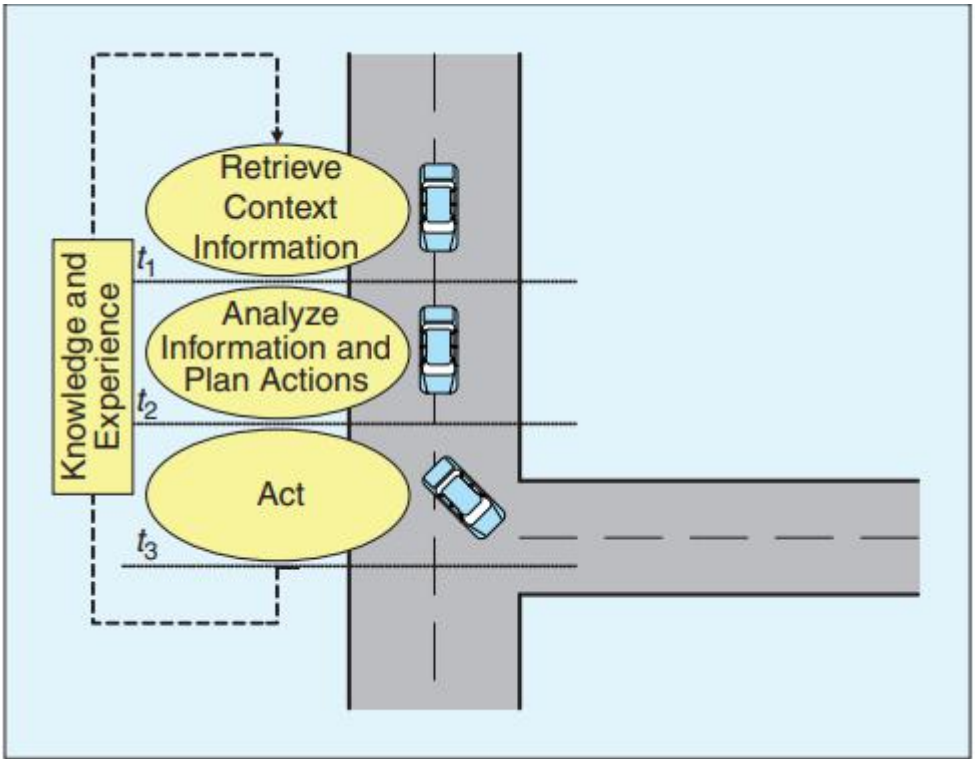


图 1-认知系统的操作

在认知网络原则的前提下，可以通过设计通信基础设施来对抗复杂性。

如图所示，放置在车内的认知系统的操作可以反映在反馈外观上。系统在时间 t1 检索上下文信息，可能是关于交通、相邻车辆的速度等的信息。通过对这些信息的分析（在时间 t2），同时考虑到自身的偏好、目标和策略，系统（在时间 t3）决定其动作（例如。，向驾驶员发出指令以改变车辆方向）。系统的输出存储在一个知识数据库中，该数据库可能只是一个供将来参考的矩阵。这意味着该系统将跟踪其行动，以便从其影响中吸取教训，为今后的决策提供便利。这在导致认知的机器学习过程[16]中重复。

总的来说，可重构和认知系统朝着最有前途的方向和技术发展，从某种意义上说，它们消除了从业务角度产生的任何潜在限制，并提供了实现真正端到端连接远景的手段。无线传感器网络

无线传感器网络

如前所述，无线传感器网络[13]、[14]构成了 B3G 无线世界的一部分。这样的网络可以包括数百个基于小电池工作的节点，因此，其生存能力可以依赖于代表其节点的资源消耗。无线传感器网络可能存在于需要进行温度、压力、湿度和速度等特定测量的区域。每一个传感器（节点）接收到一个触发器，它就将相关信息转发给整个网络（WSN）。其他传感器（节点）接收该信息并不断将其转发到一个或多个汇聚节点（即，能量限制较少、处理能力较大的节点）处，从而承担收集任何重要信息的任务（还将其进一步处理到外部网络）（见图 2）。

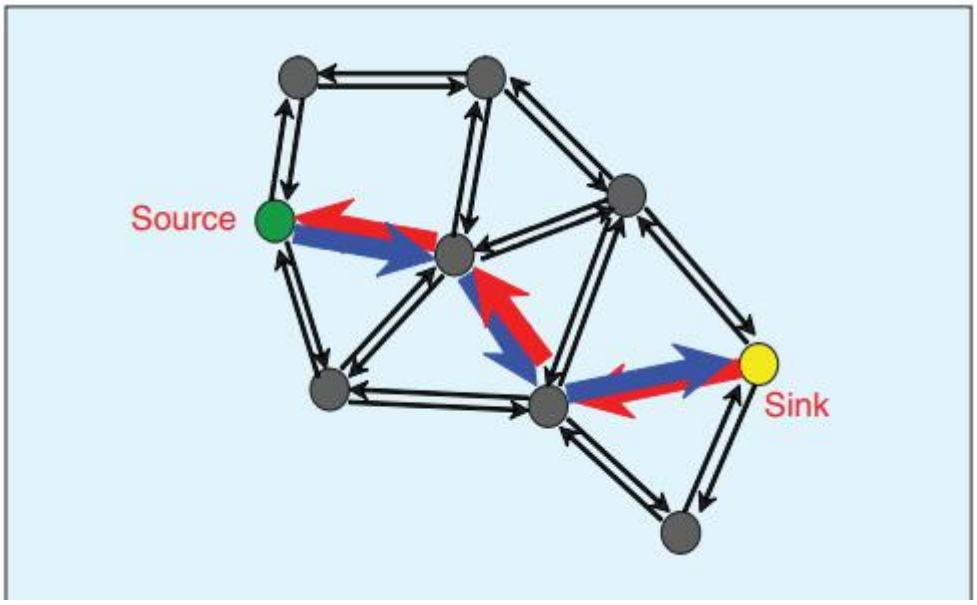


图 2-无线传感器网络中的信息传输

信息在无线传感器网络中的转发方式可能会有所不同，因为有许多用于分组路由的选项。此外，节点可能（有意地）移动，而一些节点可能由于能量问题而停止运行。这使得信息传输成为一个棘手的问题。

一般来说，无线传感器网络有许多令人兴奋的应用。在这方面，下一节将展示无线传感器网络如何与认知系统一起构成智能交通系统的一部分。

拟议功能概述

这篇文章的目的是提出管理功能的设计，包括认知通信系统，以利用（集体）的情报，通过在众多车辆之间的信息交换，在一个特定的附近积累。

一般来说，这将通过车辆动态形成的传感器网络来实现，这些车辆位于特定的地理范围内，并且能够在不同的网络节点（车辆）之间进行通信。这将导致网络车辆之间的重要信息交换。这些信息不仅可以在减少交通拥挤方面提高运输质量，而且在自动减少事故风

险和紧急情况方面也很重要。该功能在图 3 中以较高的级别显示，并在下一节中详细介绍。

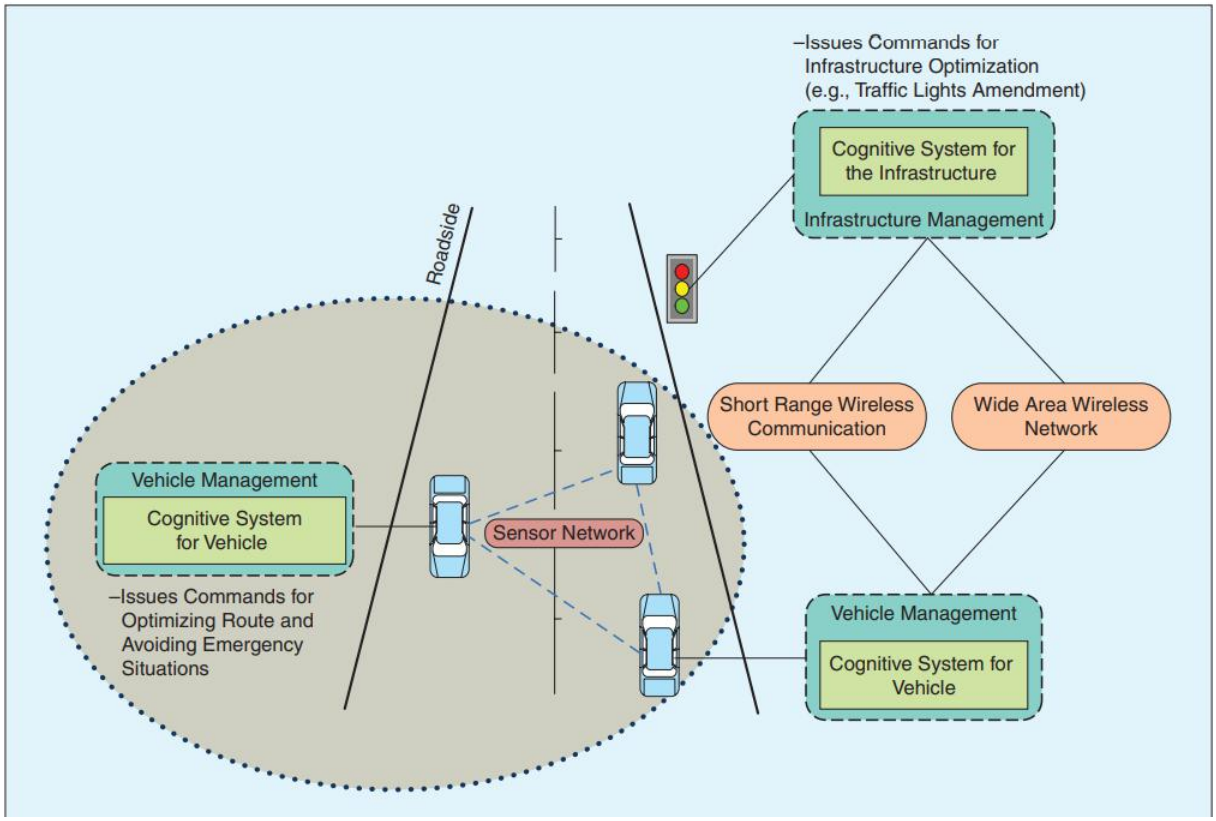


图 3-高级的功能视图。

车辆以及整个交通基础设施的智能管理功能的最重要目标是提高交通的效率和安全水平。在这方面，在设计和开发过程之前需要解决几个横向要求：

- 环境意识：识别当前环境，帮助车辆动态、安全、快速地适应环境，从而提供尽可能高的质量水平
- 个性化：不仅支持各种类型的车辆/驾驶员，而且还提供针对不同驾驶员的个性化解决方案
- 支持普适计算：支持传感器、自组网实体以及所有应用领域的局域网的存在和运行
- 始终最佳连接性：用于提供无缝网络访问，以便在异构环境中实现功能的运行
- 与备用 RAT 协作：实现功能的无缝操作
- 可伸缩性：提供不同层次的解决方案的能力，能够根据具体需求以协作或自主的方式进行操作。

车辆传感器能够在相邻车辆之间临时形成网络，以允许通信和信息交换。下一节将介绍从上述车辆管理认知功能需求分析中得出的解决方案。

组件架构和描述：拟议功能的体系结构如图 4 所示。如图所示，该功能包括以下补充组件。

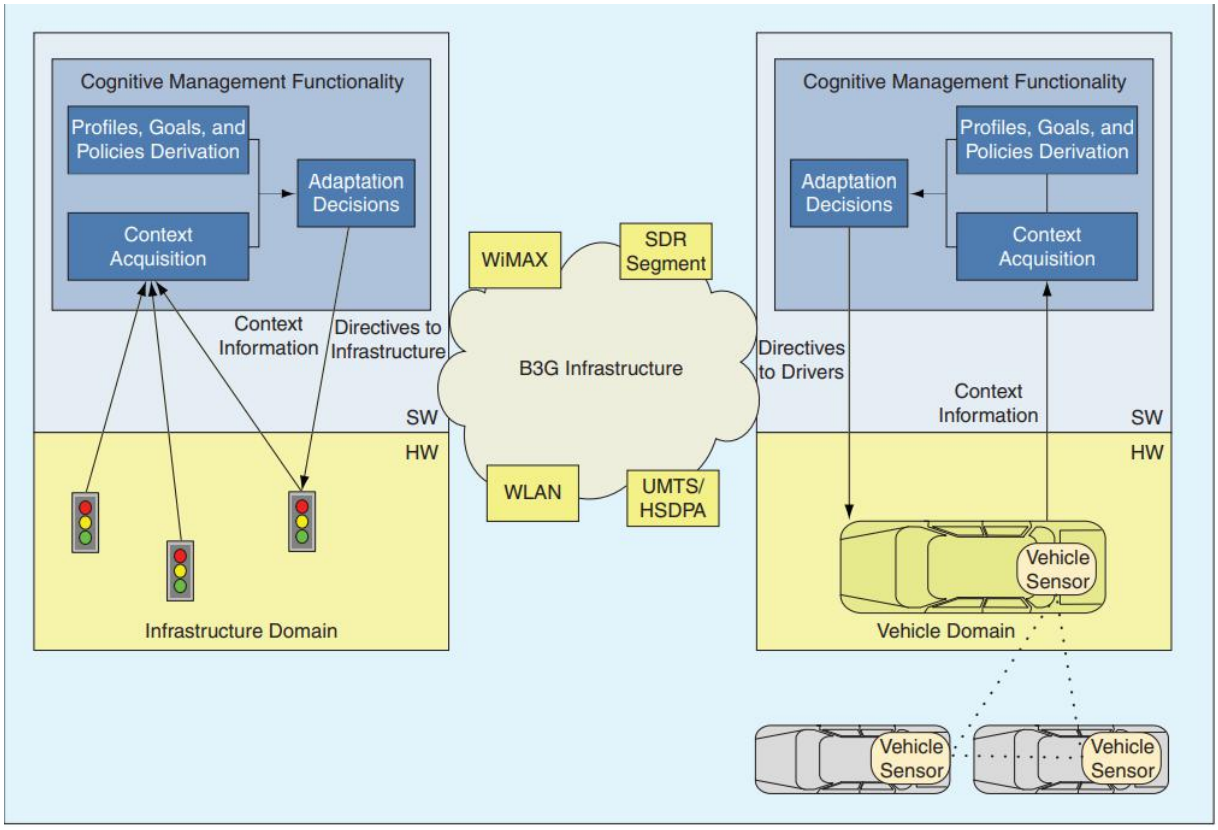


图 4-建议功能的体系结构。

车辆传感器和无线传感器网络

车辆传感器能够在相邻车辆之间形成无线传感器网络（WSNs），以允许通信和信息交换。一般来说，一辆车可以包括几个传感器。它们之间交换的信息可以分为高级数据和低级数据。

高级数据

目标和策略旨在从端到端的角度最大化决策的性能、安全性、可靠性和稳定性。此类信息包括关于拥堵程度的知识、关于潜在紧急情况的警报、驾驶员行为特征、车辆总体状况（良好、正常和不良）和巡航行为特征、道路状况信息（如打滑），邻近车辆及其巡航行为的信息，以及位置（山路、城市道路等）的一般知识。

低级数据

这类信息包括车辆的准确位置（它们之间的距离）、速度和方向、制动距离和加速度等信息。此外，还包括有关驾驶员的信息，如驾驶员简介、驾驶习惯、能力和偏好（政策）。所得到的信息可用于驾驶员当前任务的推断和驾驶员建模；i、 例如，常见的驾驶相关任务可通过对车辆-驾驶员交互作用的传感器测量值（例如，操作脚踏板、换档、按下仪表板上的按钮或查看后视镜）以及车辆状态信息（例如，位置、燃油油位和发动机状态）的解释来识别。这一过程是在不干扰司机的自然行为的情况下进行的。重点在于识别驾驶员可能从事的短期任务。

一般来说，传感器需要决定如何处理车内数据、发送哪些聚合数据、多久发送一次等。传感器测量采用专门的推理技术以分层方式进行处理，从而在不同的抽象层次上产生有关车辆-驾驶员交互的信息。

车辆认知管理功能

车辆认知管理功能（V-CMF）输入包括从传感器和无线传感器网络获取的关于车辆状态（例如其速度、方向、相邻车辆的位置、方向和速度）和路侧信息（例如路况）的上下文信息，拥堵程度和潜在紧急情况），以及交通信号灯和路标状况。此外，输入还包括有关驾驶员个人资料的信息。为此，从解释的驾驶员监控数据中推断出一组预定义的驾驶员状态（该信息也从传感器中检索）。此外，还探讨了计划识别技术来获取驾驶员的状态和行为。这意味着驾驶员和车辆之间的交互序列、关于驾驶员身体状况的原始信号（眨眼频率、眼睑张开、头部移动、轮廓、操作脚踏板、仪表板上的按钮和方向盘活动），车辆状态信息也以面部驾驶员识别的形式获取，这允许检测不同驾驶风格之间的差异。最后，司机的目标，优先事项和政策也包括在内。目标和策略旨在从端到端的角度最大化决策的性能、安全性、可靠性和稳定性。

一般来说，传感器需要决定如何处理车内数据、发送哪些聚合数据、多久发送一次等。传感器测量采用专门的推理技术以分层方式进行处理，从而在不同的抽象层次上产生有关车辆-驾驶员交互的信息。

车辆认知管理功能

车辆认知管理功能（V-CMF）输入包括从传感器和无线传感器网络获取的关于车辆状态（例如其速度、方向、相邻车辆的位置、方向和速度）和路侧信息（例如路况）的上下文信息，拥堵程度和潜在紧急情况），以及交通信号灯和路标状况。此外，输入还包括有关驾驶员个人资料的信息。为此，从解释的驾驶员监控数据中推断出一组预定义的驾驶员状态（该信息也从传感器中检索）。此外，还探讨了计划识别技术来获取驾驶员的状态和行为。这意味着驾驶员和车辆之间的交互序列、关于驾驶员身体状况的原始信号（眨眼频率、眼睑张开、头部移动、轮廓、操作脚踏板、仪表板上的按钮和方向盘活动），车辆状态信息也以面部驾驶员识别的形式获取，这允许检测不同驾驶风格之间的差异。最后，司机的目标，优先事项和政策也包括在内。目标和策略旨在从端到端的角度最大化决策的性能、安全性、可靠性和稳定性。

知识和经验

决策需要通过学习能力来加强，以便加快和提高必要的适应（重新配置）行动的效率。特别是，算法通过结合基本的学习技术（如模式匹配和上下文识别）丰富了知识特征，这些技术有助于系统将当前上下文情况与过去的情况进行比较，并确定可以实施的已经应用的解决方案。但是，也可以采用更快、更有效、更稳定的学习策略。成本因素也通过最小化电信网络中的总负载来解决。

应该注意的是，分布式上下文获取和不同程度分布的决策是由所提议的体系结构实现的，即，从 V-CMF 自主地或者以从 i-CMF 协作的方式。通常，上述组件相互协作，以便从各种来源生成知识，并产生对车辆和交通基础设施的有用指示。实现这一点的方法将在下一节中描述。

指示性信息流

本节旨在举例说明拟议功能的操作。在这方面，图 5 描述了展示功能组件之间信息交换的场景。

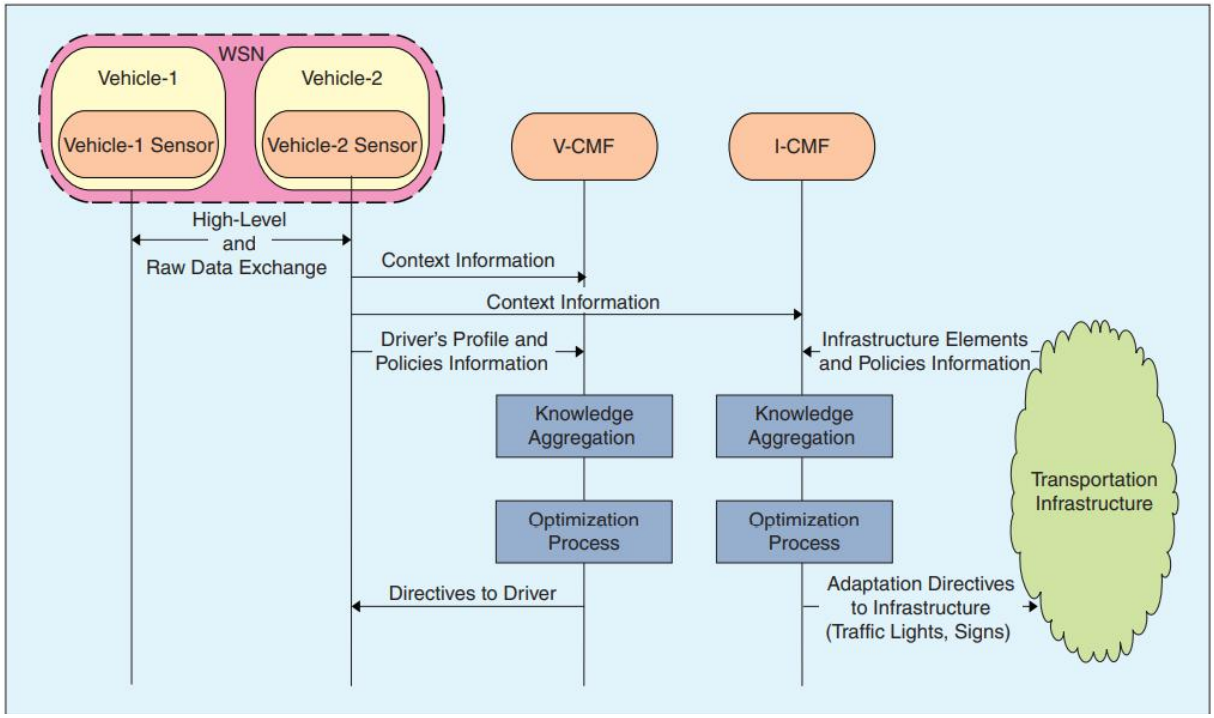


图 5-功能组件和指示性信息流

这种情况是双重的，因为它包括 1）由 V-CMF 处理的部分和 2）由 I-CMF 处理的部分。在这两个部分中，初始触发器都应该起源于无线传感器网络。无线传感器网络在其节点（传感器）之间交换信息。

在场景的第一部分中，来自 WSNs 的信息，即上下文信息以及车辆和驾驶员的概要信息、目标和策略，被传输到 V-CMF。V-CMF 收集这些信息，并将其与其知识数据库中保留的数据进行比较，以便将当前的情况（以及所作的相关决定）与过去的情况进行比较，从而在作出任何决定之前利用过去的知识和经验。然后，V-CMF 运行一个优化过程，如前所述，以便做出决策并向驱动程序发出任何必要的命令（指令），这在上下文处理中是有价值的。

车辆智能管理功能的最重要目标是提高车辆的效率和安全性。

在场景的第二部分，无线传感器网络再次向 I-CMF 发送关于当前上下文情况的信息。这里的一个主要区别是，I-CMF 还从交通基础设施收集有关基础设施要素（交通灯、路标等）和较大路段（路侧）状况的信息。更重要的是，I-CMF 从运输当局收集关于概况、目标和政策的信息，这些信息构成了 I-CMF 需要考虑的规则。然后，I-CMF 将收集到的信息与过去面临的环境和相关决策进行比较。最后，I-CMF 运行它自己的优化过程，并产生关于基础设施配置的决策，以优化某些标准，例如资源消耗、事件解决和成本最小化。

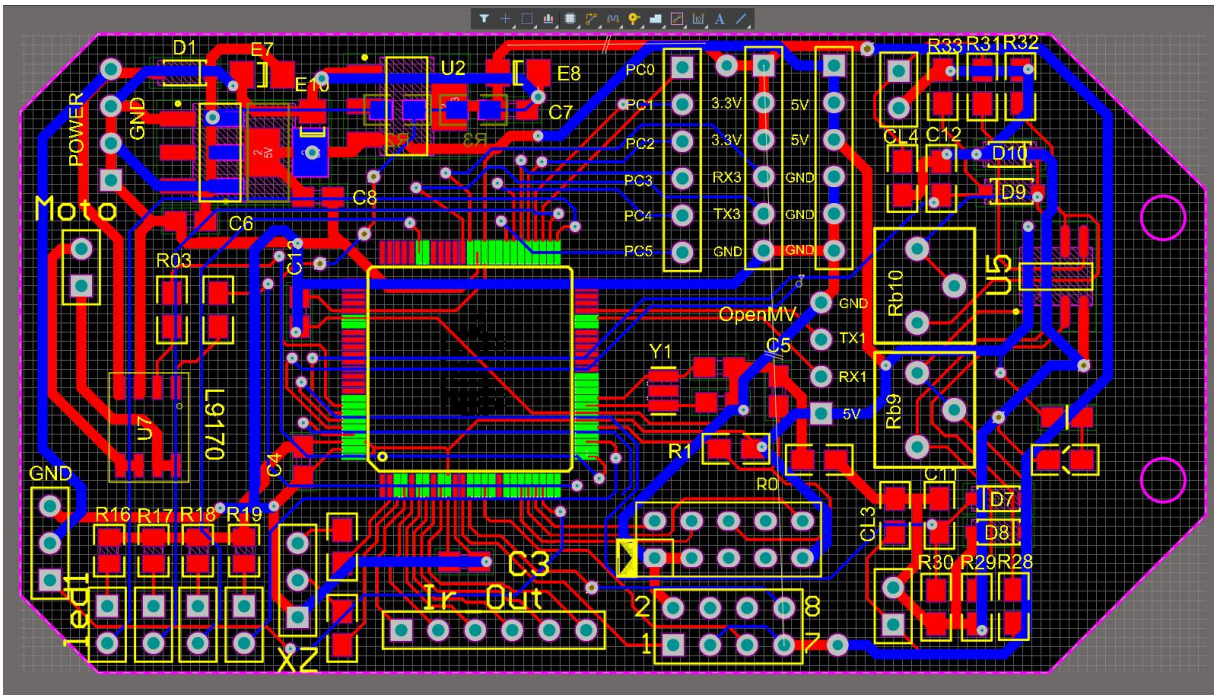
结论

无线通信的最新趋势设想了一些技术，这些技术能够在规划未来行动时利用以前与环境交互的知识。这些进步促进了交通网络中各种类型信息的无缝集成，使驾驶员受益，并提供了多种创新应用。在这方面，本文综述了一种基于认知系统和无线传感器网络的智能交通系统。提议的功能能够为驾驶员以及交通基础设施的组成部分（如红绿灯）提供有价值的指示，以满足更高效和更安全的流动性。

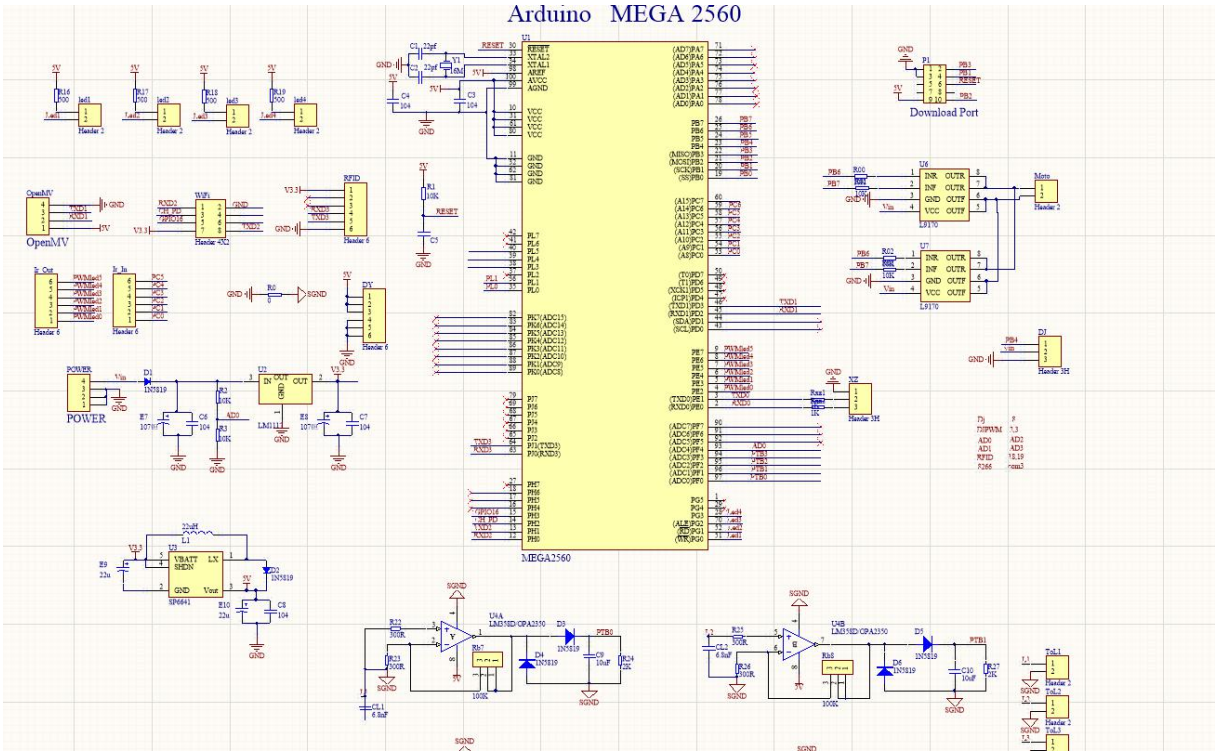
这项工作可以通过对所提议的功能组件之间的接口的详细分析来发展，以彻底描述它们之间交换的信息。此外，我们未来的部分活动将致力于在认知管理功能组件的上下文中使用特定的优化算法，以展示其有效性。

附录 C 电路图

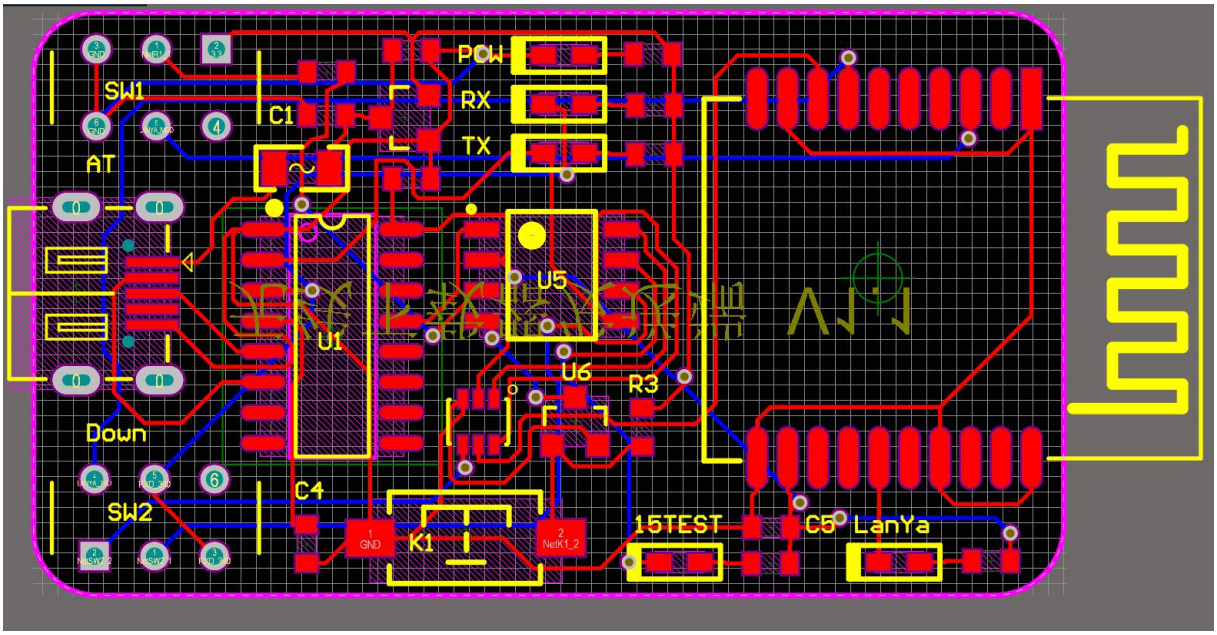
小车主控板 PCB 图



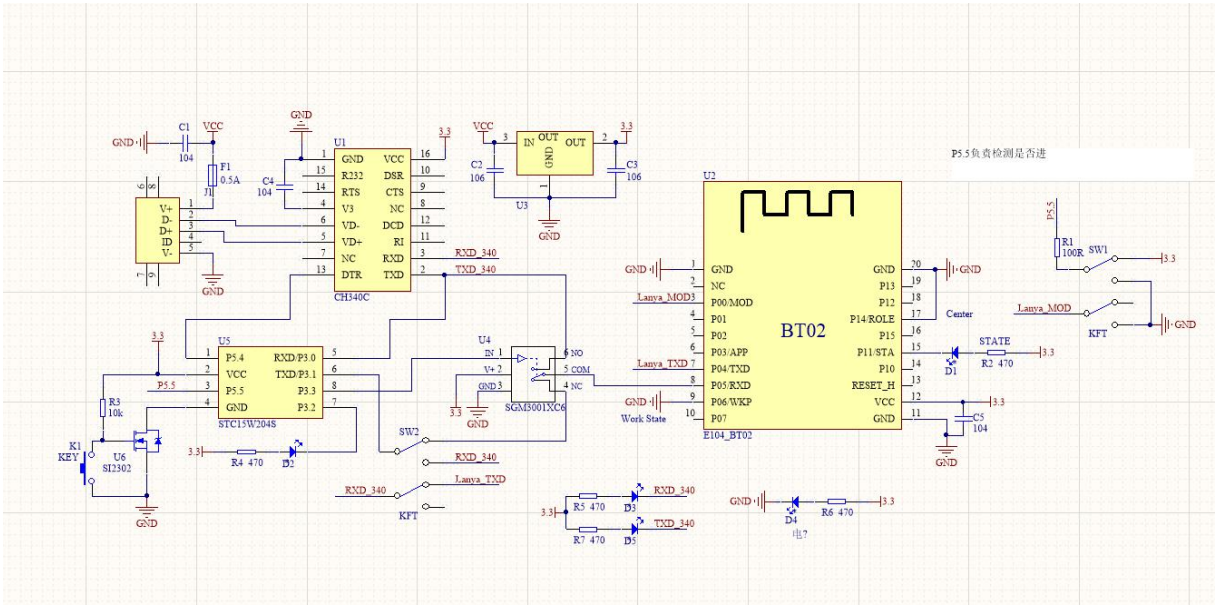
小车主控板电路原理图



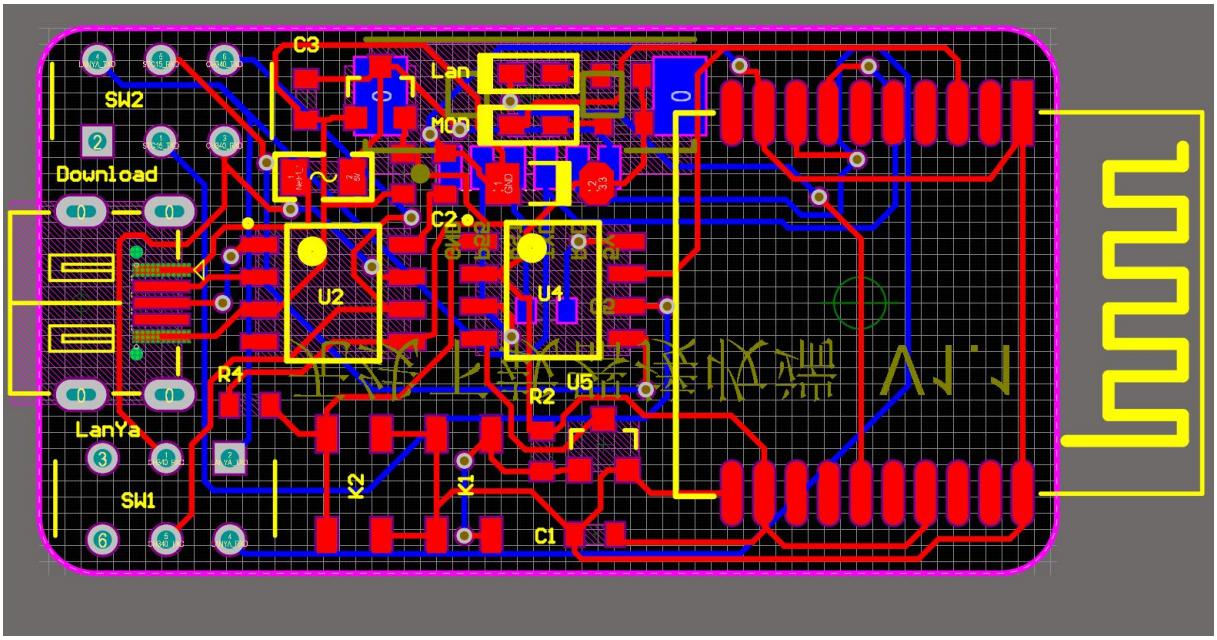
无线下载器下载端 PCB 图



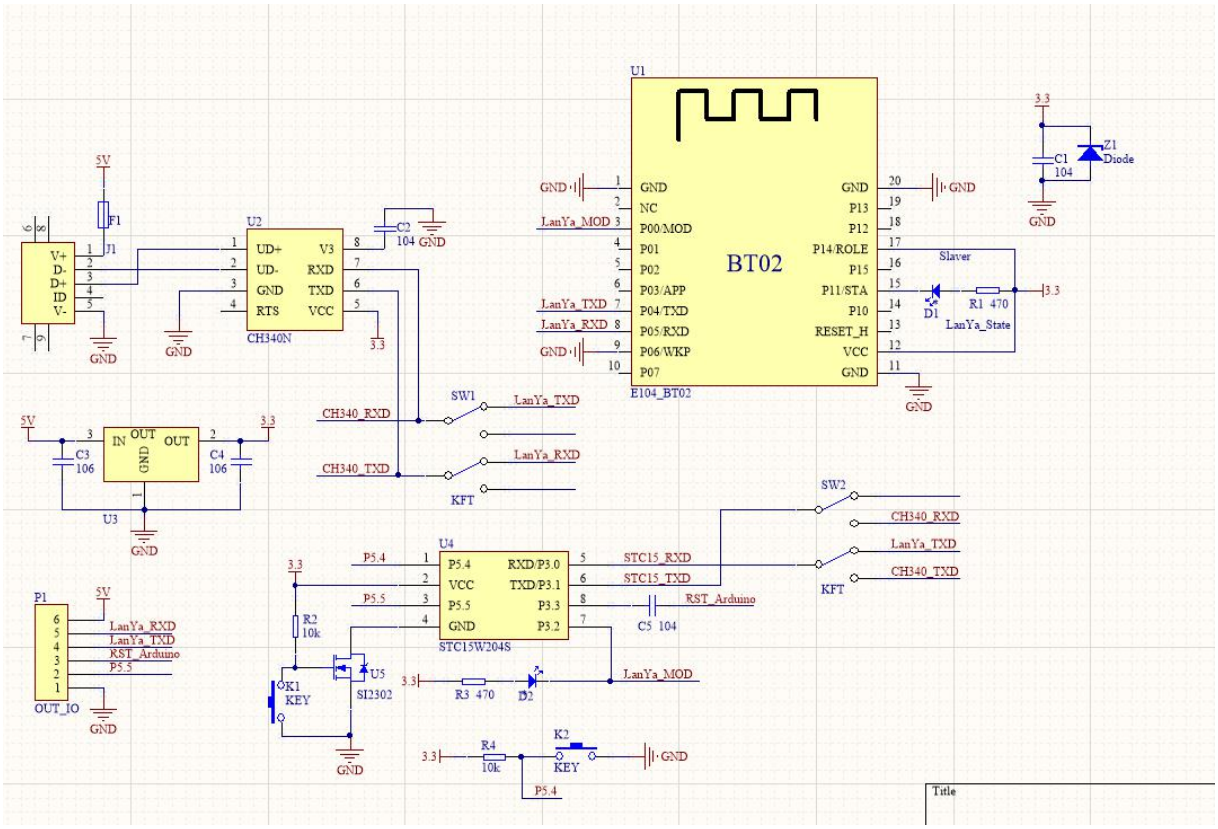
无线下载器下载端电路原理图



无线下载器接收端 PCB 图



无线下载器接收端电路原理图



附录 D 源程序

Arduino 端程序

Car_np

```
#include <MsTimer2.h>
#include <Servo.h>
#include <SPI.h>
#include <MFRC522.h>

//取消宏定义注释进入相应的测试
#define Car_GO_All        //整体程序运行
//#define Test_Motor     //测试电机，默认正转
//#define Test_Diangan_Value //Wifi 发回电感值
//#define Test_Card_Read    //测试读卡模块工作是否正常

#define SS_PIN     37
#define RST_PIN    36
MFRC522 mfrc522(SS_PIN, RST_PIN); // Create MFRC522 instance.

#define MOTOR1 13     //  电机 INR
#define MOTOR2 12     //  电机 INF

int Ind1 = A3;         //电感 1    Left
int Ind2 = A2;         //电感 2    Right
int Battery_AD = A4; //电池电压采集

Servo myservo;

void Interrupt_Init(); //中断初始化

int Timer_Count = 0;   //中断计数变量
char Time_12ms_Flag = 0;
char Time_24ms_Flag = 0;
char Time_60ms_Flag = 0;
char Time_480ms_Flag = 0;
char Time_4800ms_Flag = 0;
char Time3s_Flag = 0; //3s 定时标志位，注册失败检测
int Time3s_Count = 0; //3s 计数变量

extern unsigned char Car_Number;                  //使用的小车编号,换车时修改此处,“1”对应 1 号车
extern unsigned char Server_mid_value[25]; //舵机中值调整数组，分别对应各辆车
int Get_Light_State[8]={326,389,221,265,110,177,1320,47};   //红绿灯检测点记录数组

extern char Red_Light;         //检测到红灯则使能，为 1
extern char Barrier_Stop;   //检测到前方有障碍则使能，为 1

extern unsigned long Battery_Value;
extern unsigned long Battery_Num;

char Card_Read_Flag = 0;     //读卡测试标志位

char Go_Direction[20] = {3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3};   //行走方向暂存数组，0 为直行，1 为左转，2 为右转，3 为停止
char Car_Go = 0;             //0  运行，1  停止

unsigned char Car_Speed = 165;   //运行速度
unsigned char Car_Speed_Receive;

char Battery_Qity = 0;   //小车电量:0~5,表示电量 0%~100%
```

```
char Go_Flag = 0;//
char Stop_Flag = 0;

long Card_Data = 0;          //卡片十进制数据
long Last_Card_Data = 0; //上次卡片数据记录

char Turn_Flag = 0;      //换道标志位，为 1 使能
long Turn_Counter = 0; //打角延时计数，单位为 12ms

void setup() {
    SPI.begin();              // Init SPI bus
    mfrc522.PCD_Init(); // Init MFRC522 card
    //Traffic_IO_Init();   //Openmv 信号口初始化
    //Battery_Init();        //电池电压采集初始化
    Motor_Init();            //电机控制引脚初始化
    pinMode(Ind1, INPUT);     pinMode(Ind2, INPUT); //set inductance
    Interrupt_Init();
    myservo.attach(10);   myservo.write(Server_mid_value[Car_Number-1]);   // 70 ~ 110
    delay(50);

  #ifdef Car_GO_All       //小车程序整体运行
    Wifi_Initialize(); //Wifi
    Car_Confirm();         //小车注册
  #endif

  #ifdef Test_Motor       //测试电机
    noInterrupts();
    Motor_Turn_Test();
  #endif

  #ifdef Test_Diangan_Value       //Wifi 发回电感值
    Wifi_Initialize();   //Wifi
  #endif

  #ifdef Test_Card_Read       //Wifi 发回卡片数据
    Wifi_Initialize();   //Wifi
    Card_Read_Flag = 1;
  #endif
}

/**************6ms 中断服务函数***************/
ISR(TIMER3_OVF_vect) //中断中进行
{
    TCNT3 = 65161;
    Timer_Count++;
    Interrupt_3s();

    if(Timer_Count%2 == 0)     {Time_12ms_Flag = 1;}
    if(Timer_Count%4 == 0)     {Time_24ms_Flag = 1;}
    if(Timer_Count%10 == 0)   {Time_60ms_Flag = 1;}
    if(Timer_Count%80 == 0)   {Time_480ms_Flag = 1;}
    if(Timer_Count%800 == 0) {Time_4800ms_Flag = 1;}

    if(Timer_Count == 800)     {Timer_Count = 0;}   //计数清零
}

void loop()
{
    if( Time_12ms_Flag == 1 )
    {
        Time_12ms_Flag = 0;
        if( Turn_Flag == 1 )    //执行转向函数，屏蔽 PID
        {
            if( Turn_Counter == 0 ) { Turn_Flag = 0; }   //避免主函数直接置位,导致程序卡死
```

```
            Turn_Counter--;
            if( Turn_Counter == 0 ) { Turn_Flag = 0; }
        }
        if( Car_Go == 0 )
        {
            run_filter();     //电磁信号原始值获取,滤波
            run_guiyi();      //归一化
            if( Turn_Flag == 0 ) { run_error(); }   //计算偏差 并且赋值
        }
        if( Car_Go == 2 )
        {
            if( Turn_Flag == 0 ) { myservo.write(Server_mid_value[Car_Number-1]); }
        }
    }

    if( (Car_Go != 1 || Card_Read_Flag == 1) && Time_24ms_Flag == 1 )   //24ms 任务,只要不是停车就一直识别射频卡
    {
        Time_24ms_Flag = 0;
        Get_Card_Number(); //获取卡号
    }

    if( Time_60ms_Flag == 1 )   //60ms 任务
    {
        Time_60ms_Flag = 0;
        serial2_Event(); //数据接收及发送处理
        Path_Planning(); //利用上位机数据进行路线规划

        //Traffic_Light_Dispose(); //信号灯状态处理

        //if(digitalRead(33))   { Barrier_Stop = 1; }   //障碍检测
        //if(!digitalRead(33)) { Barrier_Stop = 0; }

        if(Car_Go == 0 && Red_Light == 0)
        {
            Go_Flag = 1;
            analogWrite(MOTOR1, Car_Speed); digitalWrite(MOTOR2,LOW);     //此处更改小车速度,最大 255
        }
        if(Car_Go == 2) { analogWrite(MOTOR1, 100); digitalWrite(MOTOR2,HIGH); }   //倒车
        if(Car_Go == 1 || Barrier_Stop == 1 || Red_Light == 1)
        {
            Stop_Flag = 1;
            digitalWrite(MOTOR1,HIGH); digitalWrite(MOTOR2,HIGH); //刹车
        }
    }

    if( Time_480ms_Flag == 1 ) //480ms 任务
    {
        Time_480ms_Flag = 0;
        //Battery_Get(); //电池电量采集
        //Serial2.print(analogRead(Battery_AD));
        //Serial2.print("      ");
        //Serial2.print(Battery_Num);
        //Serial2.print("      ");
        //Serial2.print(Battery_Value);
        //Serial2.print("      ");
        //Serial2.println((int)(Battery_Qity));
          //Serial2.write(Battery_Qity);
          //Serial2.println();

        if(!(Stop_Flag == 1 && Go_Flag == 1))
        {
            if( Car_Speed_Receive != 0 ) { Car_Speed = Car_Speed_Receive; }
            if( Car_Speed_Receive == 0 ) { Car_Speed = 165; }
```

```
                }
                if( Stop_Flag == 1 && Go_Flag == 1 )   //小车启动判断,启动加速避免卡死
                {
                    Stop_Flag = 0; Go_Flag = 0;
                    Car_Speed = 190;
                }
              #ifdef Test_Diangan_Value       //Wifi 发回电感值
                ADC_Value_Send();
              #endif
            }
            if( Time_4800ms_Flag == 1 ) //4800ms 任务
            {
                Time_4800ms_Flag = 0;
                if( !(Car_Go != 1 || Card_Read_Flag == 1) )   //车停止时仍然返回数据,上位机根据此信息判断小车是否掉线
                {
                    //Position_Return();
                }
            }
        }

        /*************** 换道函数，***************/
        /************* angle_turn：转角 ************/
        /******angle_turn:向左为负数，向右为正数*****/
        /****** time_count：打角时间，单位为 12ms ****/
        unsigned char Turn_Angle_Store = 0;
        void Turn(float angle_turn,long time_count)
        {
            Turn_Flag = 1;   //标志位置位
            Turn_Angle_Store = Server_mid_value[Car_Number-1]+angle_turn;
            if(Turn_Angle_Store <= 62)    { Turn_Angle_Store = 62; }
            if(Turn_Angle_Store >= 110) { Turn_Angle_Store = 110; }
            myservo.write(Turn_Angle_Store);     // 70 ~ 110
            Turn_Counter = time_count;
        }

        //*************中断初始化************//
        void Interrupt_Init()
        {
            /********定时器中断配置*********/
            noInterrupts();
            TCCR3A = 0;
            TCCR3B = 0;
            //timer5_counter = 65536-(62500*0.01); //0.01s 定时器中断
            TCNT3 = 65161;
            TCCR3B |= (1<<CS32);
            TIMSK3 |= (1<<TOIE3);
            interrupts();
        }

        //*************读卡部分************//
        void try_key(MFRC522::MIFARE_Key *key)
        {
                byte buffer[18];
                byte block    = 4;
                byte status;

                status      =      mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A,      block,      key,
&(mfrc522.uid));
                if (status == MFRC522::STATUS_OK)
                {
                    // Read block
                 byte byteCount = sizeof(buffer);
                 status = mfrc522.MIFARE_Read(block, buffer, &byteCount);
```

```
            Card_Data = buffer[0] + 256*buffer[1];
            if( Card_Data >= 1536 ) { Card_Data = Last_Card_Data; }      //错误数据检测
            if( Card_Data != Last_Card_Data )   { Position_Return(); }   //卡片数据更新则上报位置
            Last_Card_Data = Card_Data;
        }
        mfrc522.PICC_HaltA();           // Halt PICC
        mfrc522.PCD_StopCrypto1();    // Stop encryption on PCD
}

void Get_Card_Number()
{
    MFRC522::MIFARE_Key k;

    if ( mfrc522.PICC_IsNewCardPresent()) //寻找新卡
    {
        if ( mfrc522.PICC_ReadCardSerial()) //选中其中一张卡片
        {
            for (byte i = 0; i < 6; i++) k.keyByte[i] = 0xFF;
            try_key(&k);
        }
    }
}
```

Data_Dispose

```
#include <Arduino.h>
#include <Wire.h>

#define Car_Position_1 0x01      //小车编号
#define Car_Position_2 0x0100   //小车地址码

unsigned char Read_Buf[40]; //Wifi 数据接收缓存数组
int Address_Field; //地址码缓存
int Data_Length;     //数据长度
int Node[30];          //节点数据缓存
char Node_Num = 0; //节点数量记录
unsigned char Run_Step = 0; //执行到第几个节点计数
char Status_Inquiry[4];      //状态查询指令暂存
char Register_Confirm = 0; //中心对注册确认，0 为否  1 为是

/********数据预处理函数********/
void Data_Receive()
{
    Address_Field = int(Read_Buf [2]<<8| Read_Buf [1]);   //地址码(小车为 0xxxxxxxxxxxxxxx)
    Data_Length = int(Read_Buf [4]<<8| Read_Buf [5]);     //数据长度域; 先接收数据的高位，后接收低位

    if(Read_Buf[0] == 0xFE && Address_Field == Car_Position_2) //Read_Buf[0]为起始符
    {
        switch(Read_Buf[3])
        {
            case 0x01:   //设置智能汽车参数
                Car_Speed_Receive = Read_Buf[6];
                Run_Step = 0;   //执行到第几个节点计数归零,执行新指令
                for(char i=0;i<((Data_Length-1)/2);i++)   { Node[i]=int(Read_Buf[2*i+7]<<8|Read_Buf[2*i+8]); }
                Node_Num = (Data_Length-1)/2; //收到的节点数量统计
                Node_Data_Dispose();   //处理收到的节点数据
                Serial2.write(0xFE); Serial2.write(0x00); Serial2.write(Car_Position_1); Serial2.write(0x01);    //控
制码
                Serial2.write(0x00); Serial2.write(0x01); Serial2.write(0x01); Serial2.write(0xEF);   //确认数据已
接收
                break;
```

```
                case 0x02:   //运行参数设置
                    Car_Go = Read_Buf[6]; //启停控制
                    Serial2.write(0xFE); Serial2.write(0x00); Serial2.write(Car_Position_1); Serial2.write(0x02);   //控制码
                    Serial2.write(0x00); Serial2.write(0x01); Serial2.write(0x01); Serial2.write(0xEF);   //确认数据已接收
                    break;
                case 0x03:   //状态查询
                    for(char i=0;i<Data_Length;i++)   { Status_Inquiry[i]=Read_Buf[i+6];}
                    Serial2.write(0xFE); Serial2.write(0x00); Serial2.write(Car_Position_1); Serial2.write(0x03);   //控制码
                    Serial2.write(0x04); Serial2.write(0x00); Serial2.write(0x00); //指令码
                    Serial2.write(Car_Speed);   for(char   i=0;i<(Data_Length-1)/2;i++)   {   Serial2.write(char(Node[i])); Serial2.write(char(Node[i]>>8)); }   //发送设置参数
                    Serial2.write(0x01); Serial2.write(Car_Speed); Serial2.write(0x02); Serial2.write(char(Card_Data)); Serial2.write(char(Card_Data>>8)); Serial2.write(0xEF);
                    break;
                case 0x04:   //中心对状态确认
                    Register_Confirm = Read_Buf[6];
                    break;
                case 0x05:   //中心不确认
                    break;
                case 0x10:   //设置交通灯参数
                    break;
                case 0x11:   //设置交通灯控制命令
                    break;
            }
        }
    }

    /**********实时位置上报**********/
    //上报数据:车身速度,当前位置,电池电量
    void Position_Return()
    {
        Serial2.write(0xFE); Serial2.write(0x00); Serial2.write(Car_Position_1); Serial2.write(0x05);
        Serial2.write(0x05); Serial2.write(0x00); Serial2.write(0x01); Serial2.write(Car_Speed);
        Serial2.write(0x02); Serial2.write(char(Card_Data>>8)); Serial2.write(char(Card_Data));   Serial2.write(0xEF);

        /*Serial2.print(Battery_Value);
        Serial2.print("     ");
        Serial2.print(Battery_Qity);
        Serial2.print("     ");
        Serial2.println(Battery_Num); */
    }

    /********处理收到的节点数据********/
    //char Go_Direction[20] = {};   //行走方向暂存数组，0 为直行，1 为左转，2 为右转，3 为停止
    void Node_Data_Dispose()
    {
        for(char i=0;i<(Data_Length-1)/2;i++)
        {
            Go_Direction[i] = char(Node[i]>>12);
        }
    }

    /********利用上位机数据进行路线规划********/
    void Path_Planning()
    {
        if(Go_Direction[Run_Step] == 4)   //倒车指令
        {
            Run_Step = Run_Step+1;
            Node_Num = Node_Num-1;
            Car_Go = 2;       //倒车入库专用
```

```
        }
        if(Card_Data == ((Node[Run_Step]<<4)>>4) && Node_Num > 0)
        {
            if(Go_Direction[Run_Step] == 0) { Car_Go = 0; }         //发车
            if(Go_Direction[Run_Step] == 1)
            {
                if( Card_Data == 0x34D ) { Turn(-16.5,60); }
                if( Card_Data != 0x34D ) { Turn(-15,60);}    //左转
            }
            if(Go_Direction[Run_Step] == 2)       //右转
            {
                Turn(19,60);
                //if( Card_Data == 0X347 ) { Turn(19,60); } //出库特殊处理
                //if( Card_Data != 0X347 ) { Turn(18,60); }
            }
            if(Go_Direction[Run_Step] == 3) { Car_Go = 1; }         //停车
            Run_Step = Run_Step+1;
            Node_Num = Node_Num-1;
        }
        if(Node_Num == 0)      { Run_Step = 0; }
}

/********智能汽车注册上报********/
void Car_Confirm()
{
      for(;;)
      {
          Serial2.write(0xFE); Serial2.write(0x00); Serial2.write(Car_Position_1); Serial2.write(0x04);
          Serial2.write(0x00); Serial2.write(0x00); Serial2.write(0xEF);//发送注册指令
          Time3s_Flag = 1;
          do{
                serial2_Event();
                delay(1);
            }while(!(Register_Confirm == 1 || Time3s_Flag == 0));
            if(Register_Confirm == 1) { break; }
      }
}

void Interrupt_3s()   //12ms 中断执行计时子函数,实现 3s 定时
{
      if(Time3s_Flag==1)
      {
          Time3s_Count++;
          if(Time3s_Count>=500) { Time3s_Count = 0; Time3s_Flag = 0; } //3s 计数
      }
}

/********串口数据接收函数********/
unsigned char counter = 0;
void serial2_Event() {
   while (Serial2.available()) {

      Read_Buf[counter]=(unsigned char)Serial2.read();
      if(counter==0&&Read_Buf[0]!=0xFE) return;          //第 0 号数据不是帧头
      //Serial2.write(Read_Buf[counter]);
      if( Read_Buf[counter] == 0xEF ) //检测帧尾
      {
          counter=0;      //重新赋值，准备下一帧数据的接收
          Data_Receive();   //数据处理
      }
      else
            counter++;
   }
}
```

Motor

```
#include <Wire.h>

//********小车参数修改处 *********//
unsigned char Car_Number =15; //使用的小车编号,换车时修改此处,“1”对应 1 号车
unsigned char Server_mid_value[25]={84,91,87,84,85,85,85,85,85,85,85,85,85,85,84};  //舵机中值调整数组，分别对应各辆车
float Servo_Kp[20]={5,5,5,5,5,5};   //PID 参数设置
float Servo_Kd[20]={5,5,5,5,5,5};

//*********电机初始化*********//
void Motor_Init(){
    pinMode(MOTOR1, OUTPUT);   pinMode(MOTOR2, OUTPUT);
}

//**********原始值获取**********//
uint16_t adc_max[2] = {1000, 1000};      //获取的最大值
uint16_t adc_filter[2] = {0,0};

//**********滤波**********//
#define N 8                //滑动滤波队列大小
uint16_t sum[2];       //累和

void run_filter()
{
    sum[0] = 0; sum[1] = 0;       //累和归零
    for(char i=0;i<N;i++)
    {
        sum[0] = sum[0] + analogRead(Ind1);
        sum[1] = sum[1] + analogRead(Ind2);
    }

    for (char i=0;i<2;i++)
    {
       adc_filter[i] = sum[i] / N;                          //求平均
    }
}

//**********归一化**********//
double adc_guiyi[2];                        //归一化值

void run_guiyi()
{
  float guiyi[2] = {0, 0};
  for (uint8_t i = 0; i < 2; i++)
  {
      guiyi[i] = (float)adc_filter[i] / adc_max[i];
      if (guiyi[i] <= 0.0f)      guiyi[i] = 0.01f;
      if (guiyi[i] >= 1.0f)      guiyi[i] = 1.0f;
      adc_guiyi[i] = (uint8_t)(guiyi[i] * 100);
      if (adc_guiyi[i] < 1u)    adc_guiyi[i] = 1u;
  }
 }

//**********计算偏差并赋值给舵机**********//
float car_error = 0;               //当前偏差,偏差有正有负
float car_error_last = 0;       //上次偏差
float servo_out = 0;               //float
float real = 0;

void run_error()
{
```

```
    //计算偏差
    car_error = (float)((adc_guiyi[0]-adc_guiyi[1])/(adc_guiyi[0]+adc_guiyi[1]));
    car_error = (float)(car_error * 10u);
    //PD
    servo_out = Servo_Kp[Car_Number-1] * car_error + Servo_Kd[Car_Number-1] * (car_error - car_error_last);
    //更新偏差
    car_error_last = car_error;

    //限幅(70-110)
    if (servo_out > 40)    { servo_out = 40; }
    if (servo_out < -40) { servo_out = -40; }
    real = servo_out / 2 + Server_mid_value[Car_Number-1];
    if(real <= 62)    { real = 62; }
    if(real >= 110) { real = 110; }
    myservo.write((char)(real));
}

//**********Wifi 发回电感值**********//
void ADC_Value_Send()
{
      Serial2.print(adc_guiyi[0]);
      Serial2.print("      ");
      Serial2.print(adc_guiyi[1]);
      Serial2.print("      ");
      Serial2.println(car_error);
}

//***********电机调试函数***********//
void Motor_Turn_Test()
{
      analogWrite(MOTOR1, 240); digitalWrite(MOTOR2,LOW);    //正确为正转
}
```

Traffic Light

```
#include <Wire.h>

int Red_Stop_IO = 32;         //红灯 IO PC5
int Barrier_Stop_IO = 33; //障碍 IO PC4

//**********Openmv 信号口初始化**********//
void Traffic_IO_Init()
{
      pinMode(Red_Stop_IO, INPUT);
      pinMode(Barrier_Stop_IO, INPUT);
}

//**********信号灯状态处理 **********//
char Red_Light = 0;       //检测到红灯则使能，为 1
char Barrier_Stop = 0; //检测到前方有障碍则使能，为 1
void Traffic_Light_Dispose()
{
    for(char i=0;i<8;i++)
    {
        if( Card_Data == Get_Light_State[i] )
        {
            if(digitalRead(32)) { Red_Light = 1; } //红灯 IO PC5
        }
        if( Card_Data == Get_Light_State[i]-2 || Card_Data == Get_Light_State[i]-1 || Card_Data == Get_Light_State[i]
|| Card_Data == Get_Light_State[i]+1 || Card_Data == Get_Light_State[i]+2 )
        {
            if(!digitalRead(32)) { Red_Light = 0; }
        }
     }
```

```
}

Wifi Library

#include <Wire.h>

char result = 0;   //Wifi 用标志位，为 1 则 Wifi 连接成功

void Wifi_Initialize() //使用 TCP 协议
{
      Serial2.begin(115200);
      while( Serial2.read()>= 0 ) { }   //清空接收缓存区

      //Serial2.println("AT+CWMODE=1");   //Wifi 配置
      //Serial2.println("AT+CWJAP=\"LXT2016\",\"12345678\""); //Wifi 名称及密码配置
      //Serial2.println("AT+CWJAP=\"WM_ITS\",\"WM82165211.?\"");

      for(;;)
      {
            result = Serial2.find("IP");     //查询模块配置成功返回信息
            if(result)   { break; }    delay(1);
       }

      while( Serial2.read()>= 0 ) { }   //清空接收缓存区
      //Serial2.println("AT+CIPSTART=\"TCP\",\"192.168.43.122\",5555");
      Serial2.println("AT+CIPSTART=\"TCP\",\"192.168.1.190\",5555");   //每个人电脑 IP,端口不一样，需要更改!!
      Wifi_Check();
      while( Serial2.read()>= 0 ) { }   //清空接收缓存区
      Serial2.println("AT+CIPMODE=1");
      Wifi_Check();
      while( Serial2.read()>= 0 ) { }   //清空接收缓存区
      Serial2.println("AT+CIPSEND");
      Wifi_Check();
      while( Serial2.read()>= 0 ) { }   //清空接收缓存区

      //Serial2.println("Wifi_Connect is OK");
}

void Wifi_Check()
{
     for(;;)
     {
         result = Serial2.find("OK");
         if(result)   { break; }    delay(1);
     }
}
```

OpenMV 端程序

```
import sensor, image, time
from pyb import UART

#qiu_threshold      = ((70, 100, 8, 119, -35, 123))
red_threshold      =(72, 100, 14, 63, -11, 29)
#red_threshold=(88, 100, -33, -13, -18, 11)
#green_threshold = (88, 100, -36, -9, -19, 13)
green_threshold =(62, 100, -70, -7, -22, 49)
#(81, 100, -70, -7, -22, 49) (84, 100, -39, -6, -20, 8)
#设置绿色的阈值，括号里面的数值分别是 L A B 的最大值和最小值（minL, maxL, minA,
# maxA, minB, maxB），LAB 的值在图像左侧三个坐标图中选取。如果是灰度图，则只需
```

```
#设置（min, max）两个数字即可。

# You may need to tweak the above settings for tracking green things...
# Select an area in the Framebuffer to copy the color settings.

sensor.reset() # Initialize the camera sensor.
sensor.set_pixformat(sensor.RGB565) # use RGB565.
sensor.set_framesize(sensor.QQVGA) # use QQVGA for speed.
sensor.skip_frames(200)
sensor.set_auto_whitebal(False) # turn this off.
sensor.set_auto_gain(False)#自动增益开
roi1=(62,52,31,26)
#sensor.set_windowing(roi1)
#关闭白平衡。白平衡是默认开启的，在颜色识别中，需要关闭白平衡。
clock = time.clock() # Tracks FPS.
uart = UART(3, 9600, timeout_char=10)
# 注意！与 find_qrcodes 不同，find_apriltags 不需要软件矫正畸变就可以工作。

# 注意，输出的姿态的单位是弧度，可以转换成角度，但是位置的单位是和你的大小有关，需要等比例换算

# f_x 是 x 的像素为单位的焦距。对于标准的 OpenMV，应该等于 2.8/3.984*656，这个值是用毫米为单位的焦距除以 x 方向的感光元件的长度，乘以 x 方向的感光元件的像素（OV7725）
# f_y 是 y 的像素为单位的焦距。对于标准的 OpenMV，应该等于 2.8/2.952*488，这个值是用毫米为单位的焦距除以 y 方向的感光元件的长度，乘以 y 方向的感光元件的像素（OV7725）

# c_x 是图像的 x 中心位置
# c_y 是图像的 y 中心位置

f_x = (2.8 / 3.984) * 160 # 默认值
f_y = (2.8 / 2.952) * 120 # 默认值
c_x = 160 * 0.5 # 默认值(image.w * 0.5)
c_y = 120 * 0.5 # 默认值(image.h * 0.5)

def degrees(radians):
    return (180 * radians) / math.pi
ROI=(80,30,5,5)
tag_families = 0
#tag_families |= image.TAG16H5 # comment out to disable this family
#tag_families |= image.TAG25H7 # comment out to disable this family
#tag_families |= image.TAG25H9 # comment out to disable this family
#tag_families |= image.TAG36H10 # comment out to disable this family
tag_families |= image.TAG36H11 # comment out to disable this family (default family)
#tag_families |= image.ARTOOLKIT # comment out to disable this family


while(True):


    #find_blobs(thresholds, invert=False, roi=Auto),thresholds 为颜色阈值，
    #是一个元组，需要用括号［ ］括起来。invert=1,反转颜色阈值，invert=False 默认
    #不反转。roi 设置颜色识别的视野区域，roi 是一个元组， roi = (x, y, w, h)，代表
    #从左上顶点(x,y)开始的宽为 w 高为 h 的矩形区域，roi 不设置的话默认为整个图像视野。
    #这个函数返回一个列表，[0]代表识别到的目标颜色区域左上顶点的 x 坐标，［1］代表
    #左上顶点 y 坐标，［2］代表目标区域的宽，［3］代表目标区域的高，［4］代表目标
    #区域像素点的个数，［005］代表目标区域的中心点 x 坐标，［6］代表目标区域中心点 y 坐标，
    #［7］代表目标颜色区域的旋转角度（是弧度值，浮点型，列表其他元素是整型），
    #［8］代表与此目标区域交叉的目标个数，［9］代表颜色的编号（它可以用来分辨这个
    #区域是用哪个颜色阈值 threshold 识别出来的）。
    #if uart.any():
     #    a = uart.readline().decode().strip()
       # if      a== '1' :
         #    roi_=(185,66,60,59)
         #elif a== '2':
```

```
    #    roi_=(113,84,38,34)
    #elif a== '3':
      # print('4')
  clock.tick() # Track elapsed milliseconds between snapshots().
  img = sensor.snapshot()#.lens_corr(strength = 1.8, zoom = 1.0)
  img.draw_rectangle(roi1)
# img.draw_rectangle(roi1)
  blobs = img.find_blobs([red_threshold],x_stride=2,y_stride=2,roi=roi1,merge=True,pixels_threshold=5)
  if blobs:
  #如果找到了目标颜色
      for b in blobs:
      #迭代找到的目标颜色区域
          # Draw a rect around the blob.
          #img.draw_rectangle(b[0:4]) # rect
          #用矩形标记出目标颜色区域
          #if (b[4] >= 35):

          img.draw_cross(b[5], b[6]) # cx, cy
          #在目标颜色区域的中心画十字形标记
          uart.write('1')
          print('1')
          #uart.write("w%d\n"%b[6])
          #输出目标物体中心坐标
          #print("W%d %d"%(b[5],b[6]))

  blobs = img.find_blobs([green_threshold],x_stride=2,y_stride=2,roi=roi1,merge=True,pixels_threshold=3)
  if blobs:
          #如果找到了目标颜色
              for b in blobs:
              #迭代找到的目标颜色区域
                  # Draw a rect around the blob.
                  #img.draw_rectangle(b[0:4]) # rect
                  #用矩形标记出目标颜色区域
                  #if (b[4] >= 35):

                  img.draw_cross(b[5], b[6]) # cx, cy
                  #在目标颜色区域的中心画十字形标记
                  uart.write('3')
                  #uart.write("w%d\n"%b[6])
                  #输出目标物体中心坐标
                  print('3')

  for tag in img.find_apriltags(families=tag_families): #  默认为 TAG36H11
      img.draw_rectangle(tag.rect(), color = (255, 0, 0))
      img.draw_cross(tag.cx(), tag.cy(), color = (0, 255, 0))
      #print_args = (tag.x_translation(), tag.y_translation(), tag.z_translation(),      )
      #  位置的单位是未知的，旋转的单位是角度

      #uart.write('2')
      #if tag.z_translation()<=?
      uart.write('2')
      print('2')
      #print( tag.z_translation())
  #print(clock.fps())
```