关于新生杯最后的一些感想

既然已经确定不参加新生杯的后续比赛了，那初赛报告不能白写，毕竟花了很长时间的……再次感谢三个组员陪我参加了这出闹剧，毕竟花了一个周末加几个工作日，又花了挺长时间做统一格式和图片表格标号这种细节问题，最后也弃赛了，挺不好意思的。

规则存在很多漏洞，这是毋庸置疑的，但是决赛规则肯定会修改掉很多漏洞，想投机取巧是不太可能的。不过关于底盘改装之类的，规则肯定不会禁止，收益比还很高，可以考虑作为比赛方案的重点。至于托马斯小火车这种，额……

再次仔细看当时赶时间写的报告，还是会发现很多问题的。先说明显问题，2.3.2 可能存在歧义，但是核心理念就一个，舵机+勾爪，结构越简单越好，又不上场打架怕什么，至于插图吗，当时随便选完之后加进去的；4.2.2.2 感觉明显是当时胡乱想的，没啥理论根据，但是感觉实践起来效果倒不至于很差，毕竟其他算法都难写又难算；4.2.3.1 的混控矩阵定义有问题，应当是将整体运动分解到几个电机上，报告里面的鬼知道什么乱七八糟的东西。写报告的时候心态不太好，而且当时还是太嫩了。

其他还有一些值得改进的地方。3.2.2DWA 到现在我也搞不清楚怎么回事，当时不知道为啥要写上，但是我估计，我瞎猜的，用的上，但是收益不明显，比起 4.2.2.2 来说；整个 4 大点就应该重写，虽然这种做法确实能够解决这个取物的问题，但是如果不赶时间的话还是搞个四核的小板子跑一跑 OpenCV 效果更好一点，MicroPython 还是有自己的局限性的，提到的色块识别 mpy 转 LAB 之后 findBlob，OpenCV 用 cvtColor 转 HSV，至于其他都是些常规操作。不过视觉定位倒是可以考虑 AprilTag，大幅减小光线影响的同时还能给出方位、距离等信息，而且目标是自己做的，不限制识别对象的。基于神经网络的方法就算了吧，劳民伤财效果很可能不如 AprilTag。

究竟改底盘和自动算法哪个更重要，恐怕只有比赛结果出了才能知道吧。

上海交通大學

题目：新生杯初赛设计方案

队员：席望、徐峥、胡章立、杨凯翔

时间：2020.5.17

1 方案总述

本次新生杯方案基于 SDPmini 开发平台进行改装设计，以满足设定背景的具体需求。

底盘增设抓爪装置，配合自主设计的物料箱，并设计相应的磁力固定底座，方便货物的派送和揽收。为提高小车驱动性能，方案将对系统的动力系统进行改装，以满足比赛竞速需求。

上位机采用 SDPmini 自带的运算单元，便于路径规划算法的程序的实际编写；下位机全部设计采用 STM32 作为主控，以获得充足的算力；程序设计使用 HAL 硬件抽象库，对 STM32 各系列单片机进行兼容，方便后期底盘控制与下位机方案的调整。

2 机械结构设计

2.1 结构总述

官方提供的平台转速过低，难以满足竞速的要求，因此，方案改装了小车底盘系统，更换了驱动电压与 Kv 值更高的电机，同时对底盘的供电系统进行了改造。底盘电机带有编码器，配合上小车自有的感知系统，使得路径执行的精度更高。运输装置通过机械爪与小车固连，将货物平稳运送到终点。

2.2 感知系统

平台的感知系统由激光雷达和惯性导航单元组成。激光雷达由 SDPmini 平台提供。在机械结构的设计中，激光雷达的扫描平面上不应当出现障碍物。

2.3 底盘系统

2.3.1 驱动装置结构

平台采用双驱动轮结构，两个直流有刷减速电机连接后轮提供动力，差速控制。

2.3.2 运输装置结构

货物运送的难点主要在于小车和容器之如何固连，如何一次性收纳尽可能多的乒乓球，容器如何移动以及容器如何停留在在装卸货点。

为提高容纳体积，采用长方体物料箱进行作为运输物资的载体，物料箱上部采用盖式结构，并配有自锁装置，方便应对运输过程中的紧急事件。底部采用设置高度较高，以便适应复杂路面需求，四个轮胎带动容器减小了摩擦阻力。底部配有小型磁铁，与物资点的底座相配合。

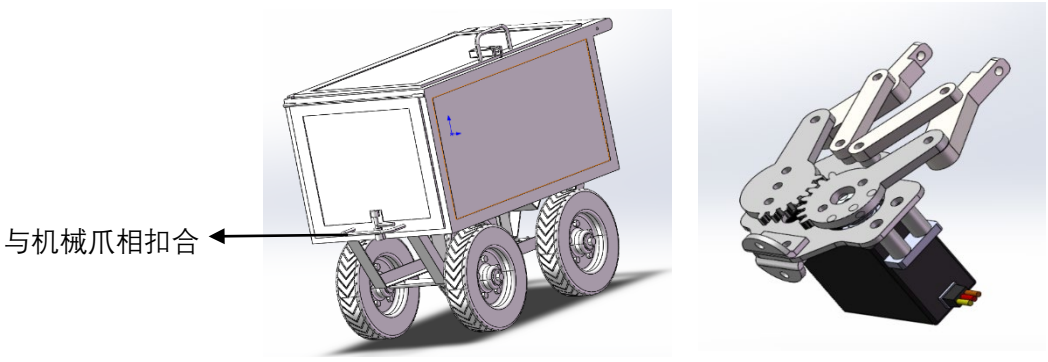

图 2.3.2 机械结构示意图

由舵机操控机械爪，通过手柄遥控，与物料箱拉杆扣合，使小车不易与物料箱脱离。用锶铁氧体磁粉与塑料复合而成的一种功能性复合材料充当运输点载体，利用其磁力强、表面光滑的特点，实现快速固定物料车。

3 路径规划算法

3.1 遥控控制

为了便于赛前的调试与应急状况的处理，系统设计时将会加入 ps2 手柄的接收器模块，接收器通过 SPI 协议与底盘通信，进行遥控控制。

代码样例如下。

首先定义 SPI 协议中片选指令

```
#define CS_L HAL_GPIO_WritePin(PORT_PS2, GPIO_PIN_PS2, GPIO_RESET);
#define CS_H HAL_GPIO_WritePin(PORT_PS2, GPIO_PIN_PS2, GPIO_SET);
```

定义发送与接收命令

```
void PS2_Cmd(uint8_t CMD)
{
 volatile uint16_t ref=0x01;
 Data[1] = 0;
 HAL_SPI_Transmit_IT(&hspi_ps2, CMD, 1);
}
void PS2_ReadData(void)
{
 CS_L;
 PS2_Cmd(Comd[0]);  //开始命令
 PS2_Cmd(Comd[1]);  //请求数据
 HAL_SPI_Receive_IT(&hspi_ps2, PS2_buf, 7);
 CS_H;
}
```

以及遥控模式判断

```
uint8_t PS2_RedLight(void)
{
 CS_L;
 PS2_Cmd(Comd[0]);  //开始命令
 PS2_Cmd(Comd[1]);  //请求数据
 CS_H;
 if( Data[1] == 0X73)   return 0 ;
 else return 1;
}
```

3.2 路径规划

3.2.1 开环控制

对于地图固定的情景，开环控制以其原理简洁、调试方法固定、执行效率高、可靠性尚可的特点在多种场景中得到了应用。在与新生杯类似的赛事中，开环控制也有成功的应用案例。东北大学在 2019 年 ROBOCON 比赛中利用开环控制取得了竞速赛全国总冠军的优秀成绩。

然而，由于开环控制完全依赖于预先调试，无法根据传感器数据进行微调，难以适

应赛场上的突发状况。一些实时的路径规划算法的引入将大大提高系统的鲁棒性。

3.2.2 路径规划总述

路径规划算法成功执行的首要条件是能够获得一个精度满足需求的场景地图。SDPmini 底盘搭载的思岚科技及 A2 Lidar 已能够满足建图需求。考虑到底盘的体积与扫图的误差，为了达到良好的运行效果，应当利用 COSTMAP 算法对扫描地图的边界进行处理，得到规划所用地图。

3.2.3 完备算法

3.2.3.1 A*算法

A*算法是路径规划中的一种常见算法。在得到充分优化后，其复杂度可降至$n\log n$，在完备算法中表现极为突出。其鲁棒性较高，对地图中障碍物分布要求较低，表现较为稳定，在多个领域中得到了广泛的应用且取得了较好的算法[1]。

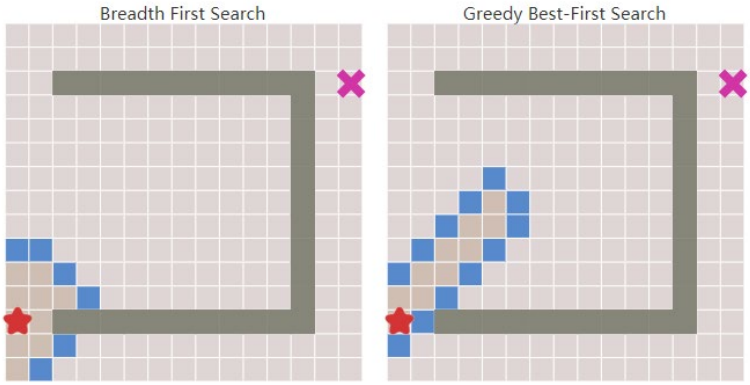

图 3.2.3 A*算法广度优先与深度优先实例

3.2.3.2 D*算法

实际操作中，地图中障碍物的分布往往是时变的。A*等静态算法难以高效地处理这种情况。由 A*算法派生的 D*算法能够高效地解决此类地图中的路径规划问题。

3.2.4 随机算法

3.2.4.1 RRT 算法

由于算法复杂度最低仅为$n\log n$，当地图扩大，目标地点距离较远时，完备算法的规划时间将大幅度增加，难以满足实时规划的需求。因此诞生了 RRT 等随机算法。RRT，即快速扩展随机树算法，是随机算法中最为基础的一种。RRT 通过随机采样的方式，每次延伸过程中有一定概率会改变延伸方向，树状遍历规划所用地图。而后利用碰撞监测等方式阻断无效分支的生长，实现高效率规划。当某一分支延伸至目标位置时，通过比较路径的长度与起始点、终点间的曼哈顿距离，判断路径是否满足规划需求[2]。

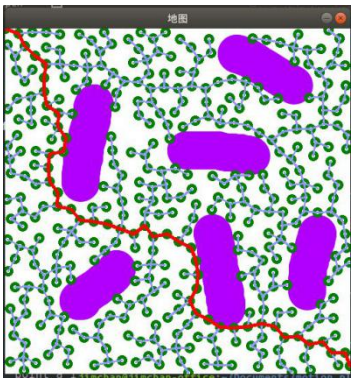

图 3.2.4 RRT 算法规划

尽管随机性提升了随机算法的执行效率，随机生长的策略在特殊地图的规划中存在一些不适用的场景。对于狭窄的缝隙，随机算法难以找到狭缝的入口，可能会导致规划的失败。

3.3 规划的执行

3.3.1 FRENET-SERRET 坐标系

FRENET 坐标系是自动驾驶领域中常用的坐标系。该坐标系使用路径坐标 s 与偏离坐标 d 替代了笛卡尔坐标系中的 x 与 y。此举将使得路径规划时的坐标意义更加清晰，也使得获得更加平滑的执行路径的过程更为容易。

3.3.2 DWA

DWA 算法全称为 dynamic window approach，其原理主要是在速度空间(v,ω)中采样多组速度，并模拟这些速度在一定时间内的运动轨迹，再通过一个评价函数对这些轨迹打分，最优的速度被选择出来发送给下位机[3]。

DWA 算法因其反应速度快，计算复杂度低，可通过线性预测得到下一时刻规划的最优解，在执行路径的规划中得到了广泛的应用。

3.4 结合官方 API 的方案综合比较

思岚科技 A 系列激光雷达已经在 rpos 类中封装了路径规划与执行的算法，实际操作中可以利用官方 API 实现路径规划与自动避障，示例代码如下

```
SlamwareCorePlatform sdp = SlamwareCorePlatform::connect(argv[1], 1445);
rpos::actions::MoveAction action = sdp.getCurrentAction();
if (action) action.cancel();
//move to location (2, 0), not on virtual track
rpos::features::motion_planner::MoveOptions options;
options.flag = MoveOptionFlag(MoveOptionFlagMilestone | MoveOptionFlagPrecise);
action = sdp.moveTo(rpos::core::Location(2, 0), options);
action.waitUntilDone();
if (action.getStatus() == rpos::core::ActionStatusError)
std::cout << "Action Failed: " << action.getReason() << std::endl;
//draw a virtual track from (0, 0) to (2, 0), then move to (0, 0) via virtual track
```

```
rpos::core::Line line(rpos::core::Point(0,0),rpos::core::Point(2,0));
sdp.addLine(ArtifactUsageVirtualTrack, line);
options.flag = MoveOptionFlag(MoveOptionFlagKeyPoints | MoveOptionFlagPrecise);
action = sdp.moveTo(rpos::core::Location(0, 0), options);
action.waitUntilDone();
if (action.getStatus() == rpos::core::ActionStatusError)
    std::cout << "Action Failed: " << action.getReason() << std::endl;
```

rpos 中封装的算法已经过充分优化，执行效果较好，同时大大节省了路径规划的开发成本。保持改装过程中整套系统对 rpos 的支持也因此成为了全部设计工作中最重要的原则。

4 物资运送方案

4.1 视觉方案

4.1.1 总述

4.1.1.1 硬件平台

在需要对图像进行处理时，常采用计算机视觉库 OpenCV，它提供了超过 500 个函数，实现了很多图像处理的算法，所有代码都经过优化，计算效率高，运行速度快，在计算机视觉的各个领域应用广泛。但基于 OpenCV 进行开发需要实现复杂的代码，并且需要相应的运算平台，所需体积过大，而且成本高，不适合在 SDP Mini 智能小车上使用。而 K210 是一款正日益流行的人工智能芯片，其中包括通用的神经网络处理器，计算能力强大且功耗低，可以高效完成基于神经网络的图像识别及图像分类任务，比如人脸识别和物体追踪等方面。因此我们选择基于 K210 芯片的二哈识图来实现所需功能。

二哈识图配有 OV2640 摄像头，内置 6 种功能，包括人脸识别、物体追踪、物体识别、巡线追踪，颜色识别及标签识别，使用简便，运行速度快。在进行物体追踪时，它可以通过不同角度对物体进行学习，使追踪效果更好。同时二哈识图的板上搭载 UART/I2C 接口，能够与运算平台进行通信。

4.1.2 传统图像处理方案

4.1.2.1 处理流程

首先将摄像头获取到的图片进行压缩，对压缩后的图将 RGB 色彩模型转换到 Lab 空间，接着在图中搜索到符合参数的块，并找到其轮廓计算长宽范围，在此过程中设置下限范围以清除噪点。找到目标色块后，根据其矩形轮廓找到中心点，再根据中心点的坐标参数，通过机械装置控制整体转动，使中心点移动到视野中央。然后由长度和宽度计算出需要前进或后退的距离，并进行移动，准确寻找到目标物体。

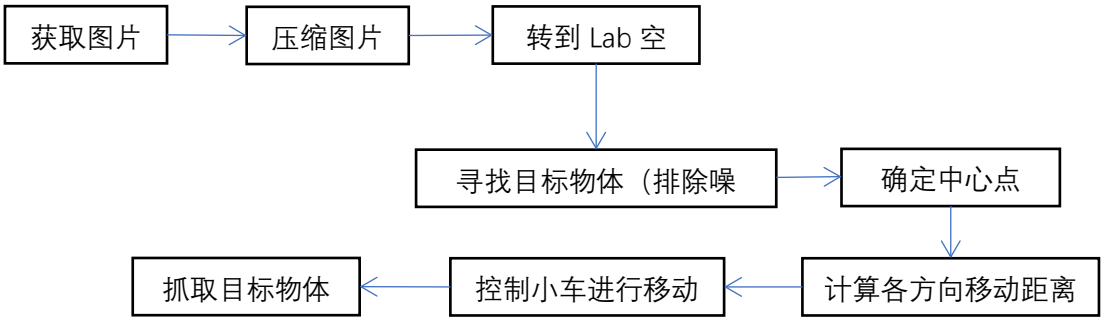

图 4.1.2　处理流程图

4.1.2.2 局限性

由于 K210 芯片采用的指令集在实现上述过程时运行速度较慢，无法快速识别目标物体，所以需要采用其它算法。

4.1.3 神经网络方案

近年来，基于神经网络的深度学习算法研究十分广泛，通过神经网络进行三维物体的识别已成常态，比如二哈识图 AI 视觉传感器通过内置的机器学习技术使其具有了人脸识别和物体识别的功能。

二哈识图采用 K210 芯片并内置 64 位 400MHz 双核 RISC-V 处理器，运行神经网络算法的速度比 STM32H743 快 1000 倍以上。经典的 YOLO 人工智能算法运行速度可以快 30 倍以上，性能十分出色，可以捕捉快速移动的物体。

下面展示出其使用效果

此处显示为学习中是因为此时开启了“边追踪边学习”的功能，可以提高学习效果。

图 4.1.3 物体追踪效果

4.2 控制方案

4.2.1 动力系统

4.2.1.1 供电系统

4.2.1.1.1 底盘传统的供电系统

底盘采用 5V 电池进行供电。通过底盘 U7 部分的 DC/DC 变换，产生可用于部分设备供电的 VCC_5V 直流电压，再通过低压差电压调节器 LM1117 产生给予其它设备供电的 VCC_3.3V 直流电压，两部分由各设备各取所需。

底盘的传统供电策略产生的电流，电压大小均难以满足，带动小车较高速度运行的需求。

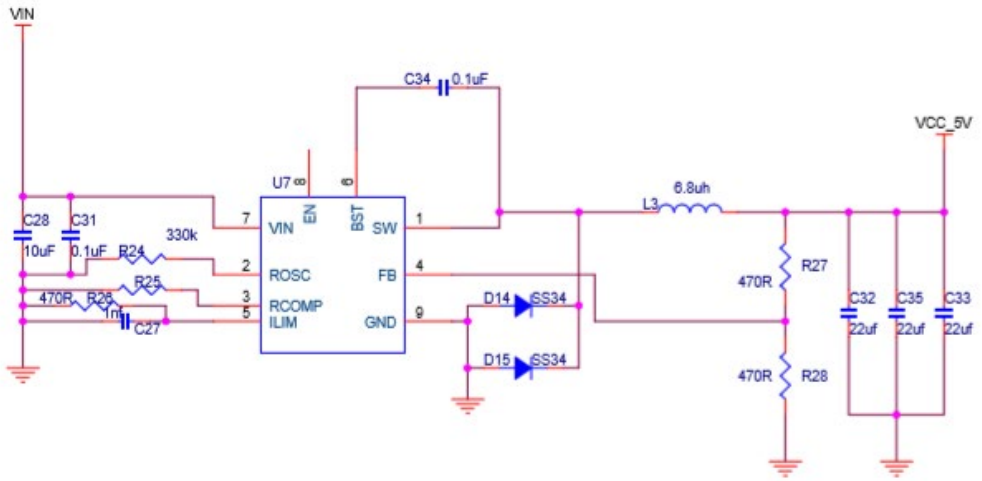

图 4.2.1-1 底盘供电电路

4.2.1.1.2 底盘的改良供电系统

采用 3S 锂动力电池 12V 代替五号电池供电，并对底盘 U7 部分加以改良。通过直流斩波器进行脉冲宽度调制，控制输出产生 VCC_5V 有效直流电压电压。同样采用压差电压调节器 LM1117 产生给予其它设备供电的 VCC_3.3V 直流电压，两部分由各设备各取所需。

底盘改良后，对比赛的竞速要求适应性更强，并添加电压测量装置，对低于$\frac{3.7\times3}{4}$的电压进行异常报警，防止过放。

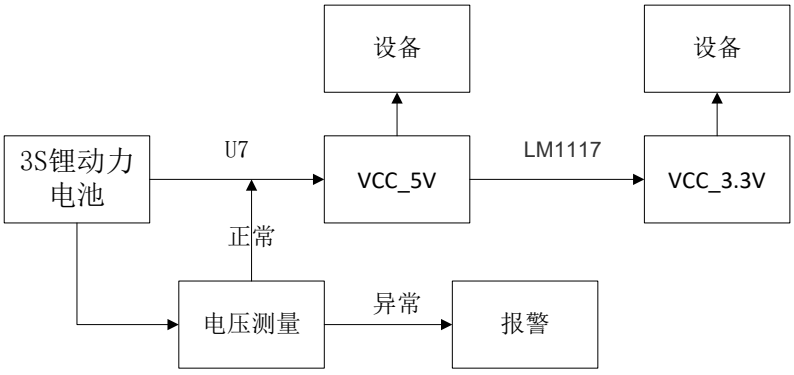

图 4.2.1-2 底盘的改良供电系统流程图

4.2.1.2 驱动系统

4.2.1.2.1 电机期望参数

查阅官方手册，发现小车车速过低，应更换转速更大的电机。

得到车轮半径为 43mm，通过计算得正常状态下每分钟转速为

$$n = \frac{250}{\pi * 43} * 60 = 110$$

最大行走速度下为：

$$n = \frac{350}{\pi * 43} * 60 = 155$$

在不考虑更换轮胎的情况下，预估电机转速不小于 400rpm.

4.2.1.2.2 电机比较和方案选择

小车要求的电机应具有高转速、高控制精度两方面的特征。针对高转速，主要排除了步进电机，对于直流无刷和有刷电机都有广泛的选择空间；而在高控制精度方面，直流无刷电机性能优良，而带上编码器的直流有刷电机也弥补了传统电机的控制精度低的缺点。因此，在两类电机满足了前两个需求的情况下，经济性成为首要指标。直流无刷电机需要搭配电调控制，在价格上普遍高于直流有刷电机。因此我们最终选择带编码器的直流减速电机代替原电机。最终敲定 ASLONG 的两款电机：JGA25-370B 与 GA12-N20B。

4.2.1.2.3 适应直流有刷电机的 H 桥式驱动电路

在确定使用直流有刷电机之后，为使电机能够实现正转、反转，改变输出的转速，需要搭载带有 PWM 波控制的 H 桥式驱动电路。

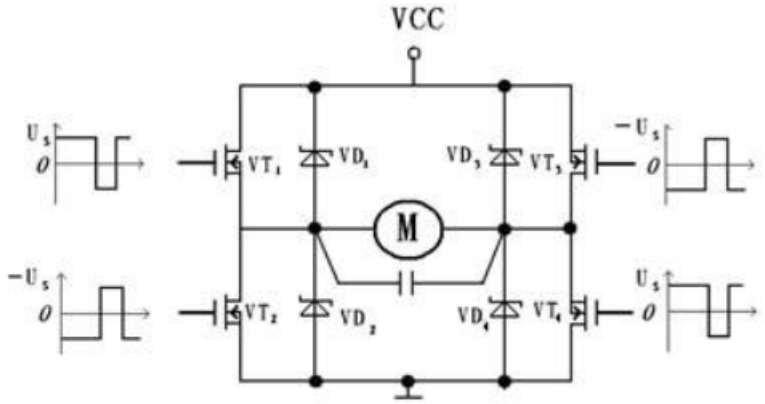

图 4.2.1-3 带有 PWM 波控制的 H 桥电路驱动电机

由单片机发出电机逻辑控制信号，通过脉宽调制，其输出信号驱动 H 桥功率电路来驱动直流电机

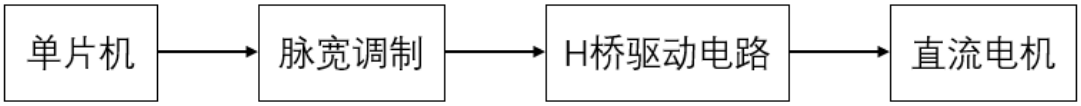

图 4.2.1-4 实现过程

电压－脉宽变换器对信号波形的调制过程，输出信号U_s的幅度不变，但脉冲宽度随U_i的变化而变化。

功放电路及其所驱动的直流伺服电机根据U_i有四种工作状态：

当U_i＝0 时，U_s的正负脉宽相等，直流分量为零，VT1 和 VT4 的导通时间和 VT2 和 VT3 导通时间相等，通过电枢绕组中的平均电流为零，电动机不转。

当U_i＞0 时，U_s的正脉宽大于负脉宽，直流分量大于零，VT1 和 VT4 的导通时间大于 VT2 和 VT3 导通时间，通过电枢绕组中的平均电流大于零，电动机正转,且随着

U_i增加，转速增加。

当U_i＜0 时，U_s的直流分量小于零，VT1 和 VT4 的导通时间小于 VT2 和 VT3 导通时间，通过电枢绕组中的平均电流小于零，电动机反转，且反转转速随着 UI 的减小而增加。

当$U_i \leqslant U_0/2$ 或$U_i \leqslant -U_0/2$时，U_s为正或负的直流信号，VT1 和 VT4 于或 VT2 和 VT3 始终导通，电机在最高转速下正转或反转。

4.2.2 混控与路径规划

4.2.2.1 阿克曼转向模型

阿克曼转向是一种现代汽车的转向方式，也是移动机器人的一种运动模式之一。

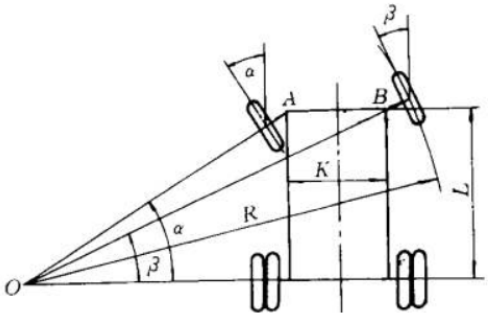

图 4.2.2-1 阿克曼转向模型

如图，α_1 α_2为前轮的左右转向角，K 为转向轮中心距，L 为轴距。由此根据几何计算得出车轮转向角公式如下:

$$\frac{1}{cot\,\alpha_1 - cos\,\alpha_2} = \frac{L}{K}$$

依据阿克曼转向几何设计的车辆，沿着弯道转弯时，利用四连杆的相等曲柄使内侧轮的转向角比外侧轮大大约 2~4 度，使四个轮子路径的圆心大致上交会于后轴的延长线上瞬时转向中心，实现无侧滑转向以保证车辆转弯形式中各车轮做纯滚动，减少行驶阻力。

4.2.2.2 路径规划模型建立

将运动轨迹放置在坐标轴上，小车必定要面临转向的情况。依照阿克曼模型设计的车辆，在转弯时，差速装置能使两个后轮驱动轮速度相等。然而官方提供的小车只有两个驱动轮，没有从动的转向轮，两轮速度可以不相等。因此，在放弃应用阿克曼模型的情况下，我们重新提出了两点间的路径规划方式，并计算所需的时间。

4.2.2.2.1 圆弧

将两目标点间路径拟合为一段圆弧，利用小车出发时的角度θ与两目标点相连后垂径，确定圆心与半径，从而得到t关于θ的函数。

当$\psi > \theta$时，运动轨迹为凸函数

通过几何关系易得：$2R\cos(\psi - \theta) = L$

小车出发前转向时间

$$t_1 = \frac{\theta}{\omega} = \frac{d}{v_c}\theta$$

沿圆弧行驶时间

$$t_2 = \frac{(\pi - 2\psi + 2\theta)R}{v_c} = \frac{(\pi - 2\psi + 2\theta)L}{2v_c\cos(\psi - \theta)}$$

到达目标点后转向时间

$$t_3 = \frac{\left|\frac{\pi}{2} - 2\psi + \theta - \alpha\right|}{\omega} = \frac{d}{v_c}\left|\frac{\pi}{2} - 2\psi + \theta - \alpha\right|$$

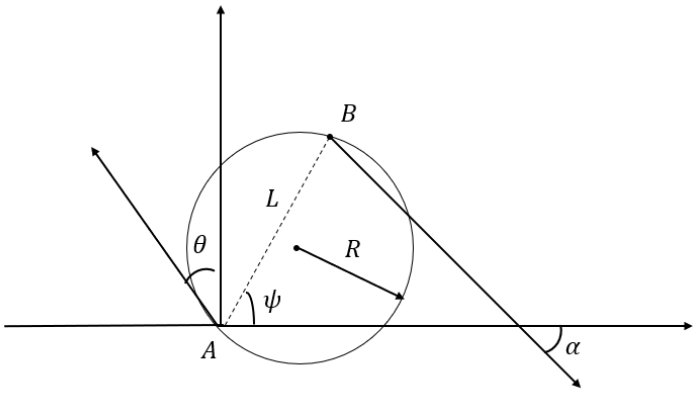

图 4.2.2-2 计算模型

当$\psi < \theta$时，运动轨迹为凹函数

$$2R\cos(\theta - \psi) = L$$

$$t_1 = \frac{\pi - \theta}{\omega} = \frac{d}{v_c}(\pi - \theta)$$

$$t_2 = \frac{(\pi + 2\psi - 2\theta)R}{v_c} = \frac{(\pi + 2\psi - 2\theta)L}{2v_c\cos(\psi - \theta)}$$

$$t_3 = \frac{\left|\frac{\pi}{2} - 2\psi + \theta - \alpha\right|}{\omega} = \frac{d}{v_c}\left|\frac{\pi}{2} + 2\psi - \theta + \alpha\right|$$

4.2.2.2.2 直线

直接将两目标点相连，分别在出发前和到达后调整小车方向。

小车出发前转向时间 $t_1 = \frac{\left|\frac{\pi}{2} - \psi\right|}{\omega} = \frac{d}{v_c}\left|\frac{\pi}{2} - \psi\right|$

直线行驶时间 $t_2 = \frac{L}{v_c}$

到达目标点后转向时间:$t_3 = \frac{|\psi - \alpha|}{\omega} = \frac{d}{v_c}|\psi - \alpha|$

4.2.2.3 路径规划模型优化

对于圆弧轨迹而言，t是关于θ的函数，那么求导之后

$$\frac{dt}{d\theta}=0$$

就能得到t的极值

在规定的出发条件下，ψ α都是定值，θ都是自变量，在去除绝对值之后有四种情况，由轨迹的凹凸性与小车末态自转方向两两组合形成。现取其中一种加以说明。

取$\psi>\theta$ $\alpha<\frac{\pi}{2}-2\psi+\theta$ 小车轨迹为凸函数，到达终点后自转方向沿顺时针

$$t=\frac{d}{v_c}\theta+\frac{(\pi-2\psi+2\theta)L}{2v_c\cos(\psi-\theta)}+\frac{d}{v_c}(\frac{\pi}{2}-2\psi+\theta-\alpha)$$

求导后得

$$\frac{dt}{d\theta}=\frac{4d}{v_c}\cos(\psi-\theta)+\frac{2L}{v_c}-\frac{(\pi+2\psi-2\theta)\tan(\psi-\theta)\,L}{v_c}=0$$

利用牛顿迭代，求出超越方程的数值解，带回 t 的表达式，就能得到圆弧轨迹下时间的最小值。

对于直线轨迹而言，在规定出发条件下同样分四种情况，由小车初态自转方向与模态自转方向两两组合形成。

取α $<\psi<\frac{\pi}{2}$ 就能得到

$$t=\frac{d}{v_c}\left(\frac{\pi}{2}-\psi\right)+\frac{L}{v_c}+\frac{d}{v_c}(\psi-\alpha)$$

实际使用时，根据目标点位置带入相应方程得到方案最优时间。对比两种路径规划模型所需的时间，选择合适的方案执行。

4.2.3 底盘控制算法

4.2.3.1 混控矩阵

当机器运动特性为两轮驱动时，通过与不同电机相连的两轮的转动为其提供前行的动力，并且通过调整两电机转速差改变其运动方向。但是驱动轮速度不能代表小车速度，所以，要推算推算电机角速度和小车速度、角速度之间的函数关系。

在曲线运动过程中，任意时刻下车轮的航向角速度相同，以后轮中心点连线的中点C(x,y)为基准点，机器ψ，电机转速与两后轮速度的关系为：

$$\frac{v_1}{R-\frac{d}{2}}=\frac{v_2}{R+\frac{d}{2}}=\omega$$

其中ω视为小车行驶的航向角速度，v_1和v_2分别为小车左、右轮速度，R为基准点到旋转中心的距离（公转半径），d为左右轮之间的距离。

在运动过程中，左右轮及基准点在做同轴运动，其角速度相同，可推出基准点线速度为左右轮线速度的一半。车体的线速度为

$$v_c=\frac{v_1+v_2}{2}=\frac{r(\omega_1+\omega_2)}{2}=\omega r$$

其中，r 为轮半径。

通过中间量R推算得航向角速度

$$\omega = \frac{d}{R}(\omega_2 - \omega_1)$$

则后轮驱动转向运动学模型为：

$$\begin{bmatrix} v \\ \dot{\Psi} \end{bmatrix} = \begin{bmatrix} \frac{r}{2} & \frac{r}{2} \\ -\frac{r}{d} & \frac{r}{d} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix}$$

4.2.3.2　PID 控制器：

按偏差的比例(P)、积分(I)和微分(D)进行控制的 PID 算法是目前应用的最为广泛的一种控制算法。该算法具有原理简单易实现，应用范围广，需要控制的参数相互独立，参数的选择比较简单等优点。由于 SDP mini 带有的 STM32F103 运算能力不强，而我们的建立的控制模型函数可以近似看作线性关系，因此采用 PID 算法。

通过路径规划模型中的过渡矩阵，由整体运动情况算出电机期望角位移，PWM 波控制电机的角速度，两者可以近似视为线性关系，角位移和角速度成线性关系。因此，可以利用双环 PID 控制方式通过占空比控制角位移。

外环中以期望角位移为输入量，控制器将控制量期望角速度输入到内环中；PID 内环中，期望角速度作为输入量，输出 PWM 波给 H 桥电机驱动电路驱动电机，编码器得到实际电机输出的角速度并反馈给控制器。实际位移被惯性导航模块反馈，通过混控矩阵转化为电机角位移，输入控制器。

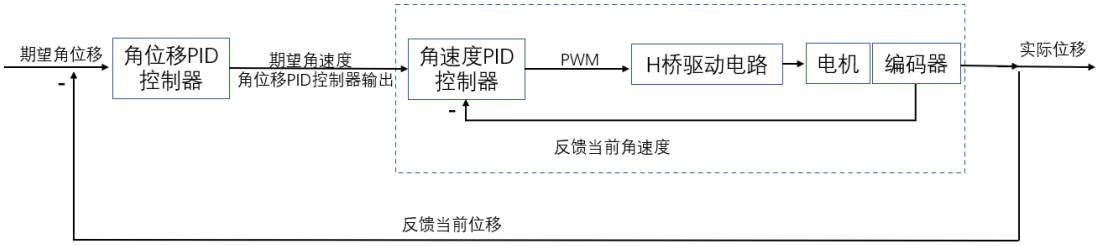

图 4.2.3　PID 算法流程

4.2.4 信息获取与滤波

4.2.4.1 快速移动过程中激光雷达的局限性

由于激光雷达扫描速度的限制，高速运行的情况下底盘很难通过激光雷达获取精度较高的位置信息。惯性导航的引入将大幅度提升系统定位的准确性。

4.2.4.2 惯性导航与滤波算法

考虑到需求对于 yaw 轴定位的精度有长时间的要求，加速度计与陀螺仪配合的传感器模式难以胜任此项工作。磁强计的引入将对 yaw 轴零飘的控制产生较大的作用，使得姿态估计更加精准，进而大幅度提升空间坐标定位的精度。

传感器的滤波采用两级滤波结构。通过低通滤波、均值滤波等低运算成本滤波方式对传感器的原始数据进行平滑处理；系统的动力学模型为线性，噪声可近似为高斯分布，故可在后级设置卡尔曼滤波器，以获得系统空间状态的最优估计。

4.2.4.3 四元数表示法

传统的欧拉角表示存在万向节锁死的弊端，且由于实数运算的偏差，欧拉角表示法的漂移较大，归一化运算所需时间较长。故姿态估计的底层算法采用四元数表示法。

利用如下对应关系，可实现四元数到欧拉角的转换，实现人机接口的功能。

$$\begin{bmatrix}\phi\\ \theta\\ \Psi\end{bmatrix}=\begin{bmatrix}\mathrm{atan2}(2(q0*q1+q2*q3,1-2(q1*q1+q2*q2))\\ \mathrm{asin}(2(q0*q2-q3*q1))\\ atan2(2(q0*q3+q1*q2),1-2(q2*q2+q3*q3))\end{bmatrix}$$

4.2.4.4 状态更新算法

对于求解微分方程的数值解，常使用龙格库塔法。此方法计算量较小，且数值解的精度较高。

采用一阶龙格库塔法，结合系统动力学方程，得到如下四元数更新方法

$$\mathrm{q0}=\mathrm{q0}+\frac{(-\omega\mathrm{x}*\mathrm{q1}-\omega\mathrm{y}*\mathrm{q2}-\omega\mathrm{z}*\mathrm{q3})*\mathrm{T}}{2}$$

$$\mathrm{q1}=\mathrm{q1}+\frac{(\omega\mathrm{x}*\mathrm{q0}-\omega\mathrm{y}*\mathrm{q3}+\omega\mathrm{z}*\mathrm{q2})*\mathrm{T}}{2}$$

$$\mathrm{q2}=\mathrm{q2}+\frac{(\omega\mathrm{x}*\mathrm{q3}-\omega\mathrm{y}*\mathrm{q0}-\omega\mathrm{z}*\mathrm{q1})*\mathrm{T}}{2}$$

$$\mathrm{q3}=\mathrm{q3}+\frac{(-\omega\mathrm{x}*\mathrm{q2}+\omega\mathrm{y}*\mathrm{q1}+\omega\mathrm{z}*\mathrm{q0})*\mathrm{T}}{2}$$

其中 T 为控制周期。

5 拓展功能

5.1 消毒喷雾

5.1.1 超声波雾化器巡线消毒

5.1.1.1 背景

疫情爆发区域人流量密集，空气环境容易被各种致病微生物污染，通过气溶胶、间接接触等传播方式导致感染发生。传统手工药物消毒方案人力成本高，工作风险大，易造成二次污染，难以满足疫区地毯式需求。因此，采用新型消毒技术做好环境卫生消毒对于抑制严峻的疫情局势至关重要。

5.1.1.2 原理

超声雾化器工作原理是将超声波振荡电能转换成机械能，使消毒液变成微小的雾粒弥散至空气中。由功率发生器产生的 1.45MHz 以上的高频电流经过安装在雾化缸里的 PZT 压电陶瓷换能器使其将高频电流换为相同频率的超声波；由换能器产生的超声波通过雾化缸中的耦合作用，穿过雾化片的透声薄膜，从而使超声波直接作用于消毒液；由声能的动力作用而产生张力波，张力波使消毒液形成雾粒，可随气流而弥散至空气中[4]。

液位传感器采用非接触电容式传感器，它首先根据电容大小与电容尺寸相关联的原理，将消毒液液位高度变化转换成电容量变化。

令消毒液的介电常数为εr，空气的介电常数ε0＝1，则液位变化时，电容器的电容变化值ΔC 与被测材料的物位高度 x 成线性关系，即

$$\frac{\Delta C}{C_0}=\frac{x(\varepsilon_r-1)}{h}$$

式中 h—电容器的总高度；C0—初始电容值[5]。

而后，电容变化值经过调制转化成电压脉冲宽度变化，单片机对电压脉冲宽度进行测量并转换成相应的液位高度进行显示，即得到液位的具体参数。

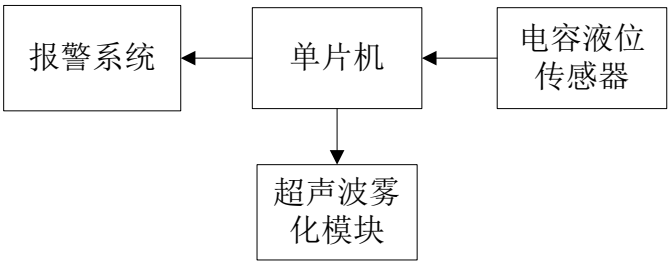

图 5.1.1-1　系统整体框图

为了提高系统的可靠性、稳定性及可维护性,该设计选用模块化思想对硬件电路进行设计,系统由单片机最小系统和其他硬件电路构成整个电控单元。

STM32 单片机是该雾化消毒设计的核心部分，实现的功能包括接收电容液位传感器信息，触发报警系统，发送指令控制超声波雾化模块。

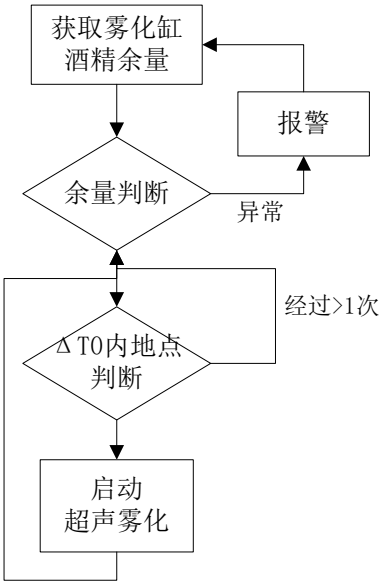

图 5.1.1-2 程序流程图

代码样例如下。

首先初始化定时器并打开定时器输入捕获功能

```
HAL_TIM_BASE_START(&htimx);
HAL_TIM_IC_Start(&htimx,C_TIM_CHANNEL);
```

而后通过测量 RC 电路充放电时间，估测电容值

```
static uint16_t TOUCHPAD_Get_Val(void){
```

```
    CAPACITOR_Reset();
    while(__HAL_TIM_GET_FLAG(&htimx,C_TIM_FLAG_CCR)==RESET)  {
        uint16_t count;
        count=__HAL_TIM_GET_COUNTER(&htimx);
        if(count>(TOUCHPAD_TIM_ARR-500))

        return count;       //time out
      }
      return HAL_TIM_ReadCapturedValue(&htimx,C_TIM_CHANNEL);
  }
```

5.2 数据交互

5.2.1 项目背景

对无人配送车在起始点的打包揽收环节，和运输过程中的投递派送环节来说，数据交互无疑是重要的一环。通过信息的传递，设备可以对环境与自身的信息进行采集、整序、分析，既方便管理者对配送工作宏观把控，掌握更全面的管理信息，又方便了技术人员实时监控设备的运行状况。这无疑有效提高物资运输过程中的安全性、可靠性，提升了运输效率。

5.2.2 通讯内容

5.2.2.1 正常状态

对配送车的位置坐标，车距和速度等具体参数实时测量并上传终端，作为物资运输的重要数据加以保留，方便管理者实施监控和进行相关分析工作；

5.2.2.2 异常状态

当车辆的实时参数超过预设阈值时，识别车辆进入异常状态，对相关车辆密切监控，并将信息反馈给管理人员，以便及时人为处理突发状况。

5.2.3 硬件基础与代码实现

为满足数据交互的需要，系统中将会增设 Wi-Fi 数传模块。

在正常状态下，车辆的传感器识别到相关数据后，由 ESP8266 通过 TCP 协议上传至服务器，方便技术人员整合和分析，管理者宏观把控。

在异常状态下，技术人员发现回传数据传输异常后，可通过 TCP 协议发送指令数据，控制无人配送车，进而方便进行紧急制动，人为干涉等应急措施。

代码样例如下。

首先是对 AT 指令所需的字符串常量的定义，三个字符串数组分别为 AP 模式初始化、STA 模式初始化、TCP 发送所需指令。

```
const uint8_t CMD_esp_ap[][45] = {
    "AT+CWMODE=2\r\n", //AP mode
    "AT+CWSAP=\"username\",\"password\",1,0\r\n",  //Start AP
    "AT+CIPMUX=1\r\n",          //muticonnection
    "AT+CIPSERVER=1,6789\r\n",
};
const uint8_t CMD_esp_sta[][25] = {
```

```
    "AT+CWMODE=1\r\n", //STA mode
    "AT+CWJAP=",      //Join AP
    "\"username\",",          //username
    "\"password\"\r\n",           //password
    "AT+CIPMUX=0\r\n",          //monoconnection
    "AT+CIPMODE=1\r\n",
    "AT+CIPSTART=\"TCP\",",          //Start connection
    "\"192.168.4.1\",6789\r\n",
};
const uint8_t CMD_esp_send[][25] = {
    "AT+CIPSEND=0,00\r\n",
    "AT+CIPSEND",
};
```

而后定义模块初始化函数

```
void init_Esp8266(){
    HAL_UART_Receive_IT(&huart_esp, esp_buf, 1);

    for(uint8_t j = 0; j < 4; j++){
        while(HAL_OK != HAL_UART_Transmit_IT(&huart_esp, cmd_esp_ap[j],
    sizeof(cmd_esp_ap[j])));
        while(!esp_flag);
        HAL_Delay(1000);
        esp_flag = 0;
    }
}
```

最后定义发送函数

```
void esp_transmit(uint8_t *data, uint8_t size){
    cmd_esp_send[0][13] = size / 10 + '0'; //update the size of data to be
transmitted
    cmd_esp_send[0][14] = size % 10 + '0';

    HAL_UART_Transmit(&huart_esp, cmd_esp_send[0], sizeof(cmd_esp_send[0]),
1000);
    HAL_Delay(200);
    HAL_UART_Transmit_IT(&huart_esp, data, size);
}
```

6 参考文献

[1] Introduction to the A* Algorithm[EB/OL]
https://www.redblobgames.com/pathfinding/a-star/introduction.html
2014-05-26/2016-06
[2] Siciliano B, Oussama K. Springer Handbook of Robotics[M]. 2007.
[3] DWA 算法分析[EB/OL]
https://blog.csdn.net/peakzuo/article/details/86487923,2019
[4] 何云，刘柏岩. 关于医用超声波雾化器性能改进的探讨[J]. 河北工业大学学报，2008，037(003):107-111.
[5] 张天航，杨超. 电容式液位计/开关升级及与 DCS 接口改进[J]. 仪器仪表用户，2019(10).