第十七届全国大学生智能汽车竞赛

技 术 报 告

学　　校：杭州电子科技大学

队伍名称：杭电百度竞速队

报告日期：2022 年 8 月 15 日

目录

1 引言

全国大学生智能汽车竞赛是以“立足培养、重在参与、鼓励探索、追求卓越”为指导思想， 鼓励创新的一项科技竞赛活动。百度完全模型组的竞赛规则把比赛分成了两部分，分别是线上赛（占总成绩的 10%）和线下赛（占总成绩的 90%）。线上赛需要使用百度 AIStudio 以及百度 Paddleseg 套件完成车道线图像分割任务。线下赛要求参赛队员使用赛曙科技制造的 I 型车模（阿克曼结构），并使用百度公司的 AI 解决方案——Edgeboard 作为上位机来读取彩色摄像头图像并进行图像处理、深度学习模型预测等工作，使用英飞凌公司的 TC 系列单片机并自行设计电路板（下位机主控板、电机驱动板等）来实施车辆的姿态控制。本文主要就线下赛所用的汽车模型的硬件与软件设计进行说明。我们需要制作一个可以完成摄像头巡线、目标检测、惯性导航等任务的模型汽车。参赛队员的目标是模型汽车需要以最快的速度从起点出发，完成一系列任务并且通过赛道元素后回到终点。在准备比赛的过程中，我们小组成员涉猎控制、模式识别、传感技术、汽车电子、电气、计算机、车辆工程等多个学科，这次磨练对我们的知识融合和实践动手能力的培养有极大的推动作用，在此要感谢清华大学，感谢他们将这项很有意义的科技竞赛引入中国；也感谢杭州电子科技大学对比赛的关注，我们的成果离不开学校的大力支持及指导老师悉心的教导；还要感谢的是和我们一起协作的队员们，协助，互促，共勉使我们能够走到今天。

2 总体思路概述

2.1 线上赛概述

此次第十七届智能车竞赛完全模型组线上赛要求参赛者利用提供的训练数据，在统一的 AIStudio 平台计算资源下，实现一个能够识别虚车道线、实车道线和斑马线具体位置和类别的深度学习模型，不限制深度学习任务。本届赛题数据集包括 16000 张可以直接用于训练的车载影像数据，官方采用分割连通域标注方法，对这些图片数据标注了虚车道线、实车道线和斑马线的区域和类别，其中标注数据以灰度图的方式存储。标注数据是与原图尺寸相同的单通道灰度图，其中背景像素的灰度值为 0，不同类别的目标像素分别为不同的灰度值。实车道线、虚车道线和斑马线类别编号分别为 1、2 和 3，0 为背景编号。本次比赛要求选手使用飞桨 PaddlePaddle2.1 及以上版本生成端到端深度学习模型。模型预测速度在 v100 显卡需达到 30FPS 及以上（预测时间包括前处理和后处理用时），视为有效，在满足速度基础上，按照精度高低进行排名。与通常的图像分割任务一样，比赛采用 mIoU 指标来评估结果。（图像分割示范样例如图 1 所示，mIoU 指标计算方法如图 2 所示，其中，C 是分类数，在本任务中是类别数，TP, FP 和 TN 表示 true positive, false positive and false negative。）

图 1：线上赛图像分割示范样例

$$\mathrm{IoU}_c = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$$

$$\mathrm{TP} = \sum_i \| M_{ci} \cdot M_{ci}^* \|_0$$

$$\mathrm{FP} = \sum_i \| M_{ci} \cdot (1 - M_{ci}^*) \|_0$$

$$\mathrm{FN} = \sum_i \| (1 - M_{ci}) \cdot M_{ci}^* \|_0$$

图 2：mIoU 指标计算方法

2.2 线下赛概述

此次第十七届智能车竞赛完全模型组线下赛要求选手使用模型汽车，在如图 3 所示的室内循环赛道中完成竞速任务，用时短者成绩优先，白色赛道铺设在铺有蓝色广告布的场地上，赛道边缘铺设有黑色胶带以及路肩。完全模型组巡线可以使用传统视觉图像处理方案或者深度学习巡线方案，同时需要完成泛行区、施工区、坡道等任务，来获得相应的减时，来提高比赛成绩。以下为各个任务描述。

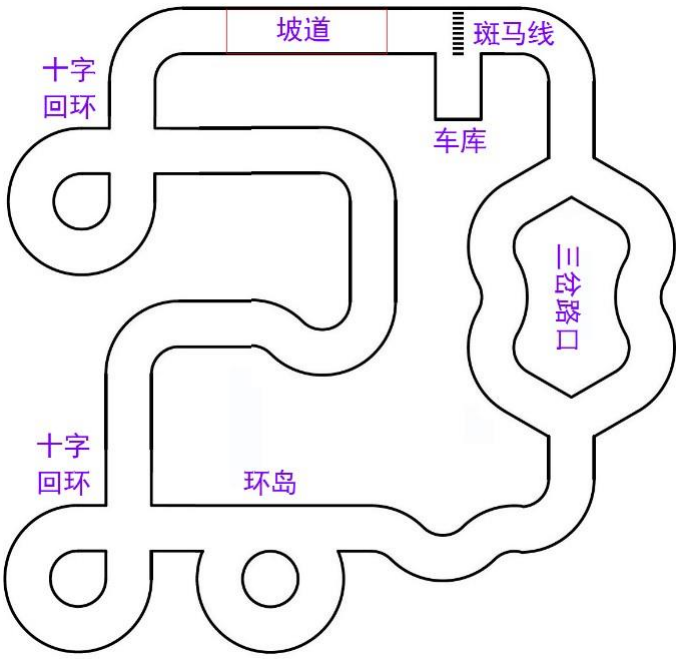
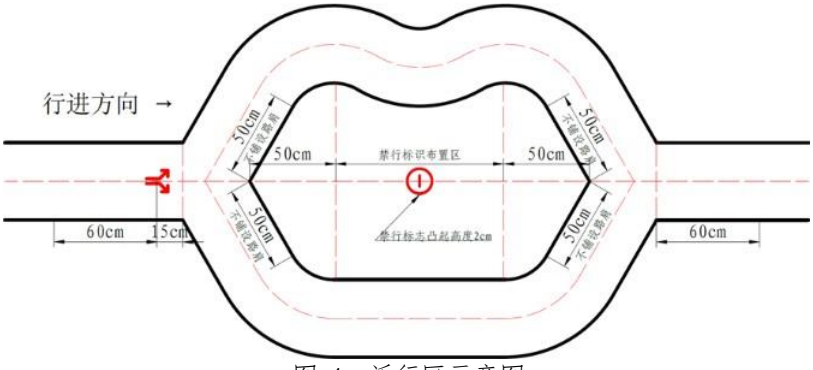


图 3：室内循环赛到示意图

2.3 泛行区任务描述

两个岔路口围成的区域为泛行区，在完全模型组别中模型汽车行驶不受三岔路两条线路内部闭合的道路边界线的限制，内部闭合的道路边界线特定区间内不铺设路肩。参赛队自行选择最优的路线从三岔路的入口行驶到出口即可。三岔路口的入口车道中心处放置有识别定位标志，表示前方为泛行区。泛行区的蓝底布区域出口和入口中心连线的指定范围内放置有禁止通行的标志（沿布置区中心连线随机放置），车辆需要绕过此标志进行通行。车辆在蓝底布部分通过泛行区，奖励减时间 5 秒，车辆撞到禁止通行的标志后通过泛行区既不惩罚也不奖励。路肩的铺设要求和标志放置如图 4 所示。


行进方向 →
50cm
50cm
禁行标识布置区
50cm
50cm
50cm
50cm
50cm
60cm
15cm
60cm
图 4：泛行区示意图

2.4 施工区任务描述

赛道中选择一段直道，设置施工绕行区域。施工绕行道路采用锥桶围成，从赛道的边缘黑线区处开始放置（入口处黑线边缘不放置），组成一条临时通行车道。车辆行驶到施工区的时需要离开常规的赛道从搭建的绕行车道通行。施工区域前方 70 厘米处(标志中心到锥桶中心线的距离)放置施工标志指示车辆前方为施工区需要进入临时道路进行绕行。施工区示意图如图 5 所示。图中每个红色圆圈代表 1 个小锥桶，实际摆设中存在锥桶中心不在同一直线以及间距也不完全相等，误差在锥桶半径范围内。车辆未进入施工绕行道路 1 次罚加时 5 秒，车辆每次成功通过绕行区奖励减时间 5 秒。

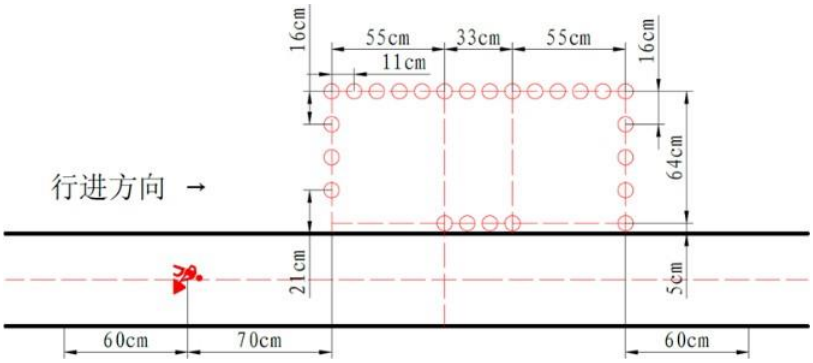


图 5：施工区示意图

2.5 坡道任务描述

基础赛道中设置有坡道，为了契合人工智能自动驾驶的技术路线，因此在坡道前方 15 厘米处贴放坡道的标志，供人工智能模型的识别，便于提前预知坡道从而进行速控操作。坡道示意图如图 6 所示。

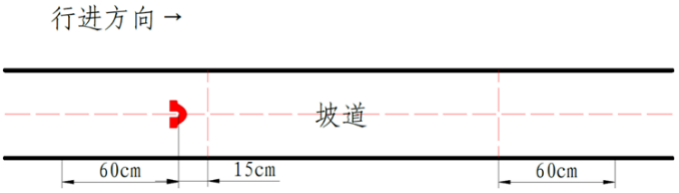


图 6：坡道示意图

2.6 总体思路概述

本次完全模型组模型汽车的工作模式为：使用摄像头拍摄赛道，在 Edgeboard 上位机读取获得 640*480 的 RGB 三通道图像，图像用于图像处理以获取赛道信息，将图像处理得到的车身偏差以及检测到任务点之后对下位机的操作信息等通过串口通信传输到下位机，再由下位机完成对电机的速度控制以及舵机的转向控制，以此来完成小车的姿态控制。此次我们组的整体方案思路示意图如图 7 所示。

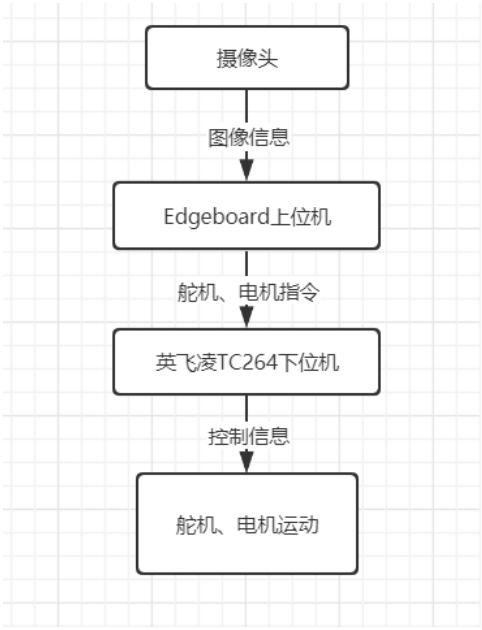


图 7：整体方案思路示意图

3　总体硬件方案及实现

3.1　百度EdgeBoard套件简介

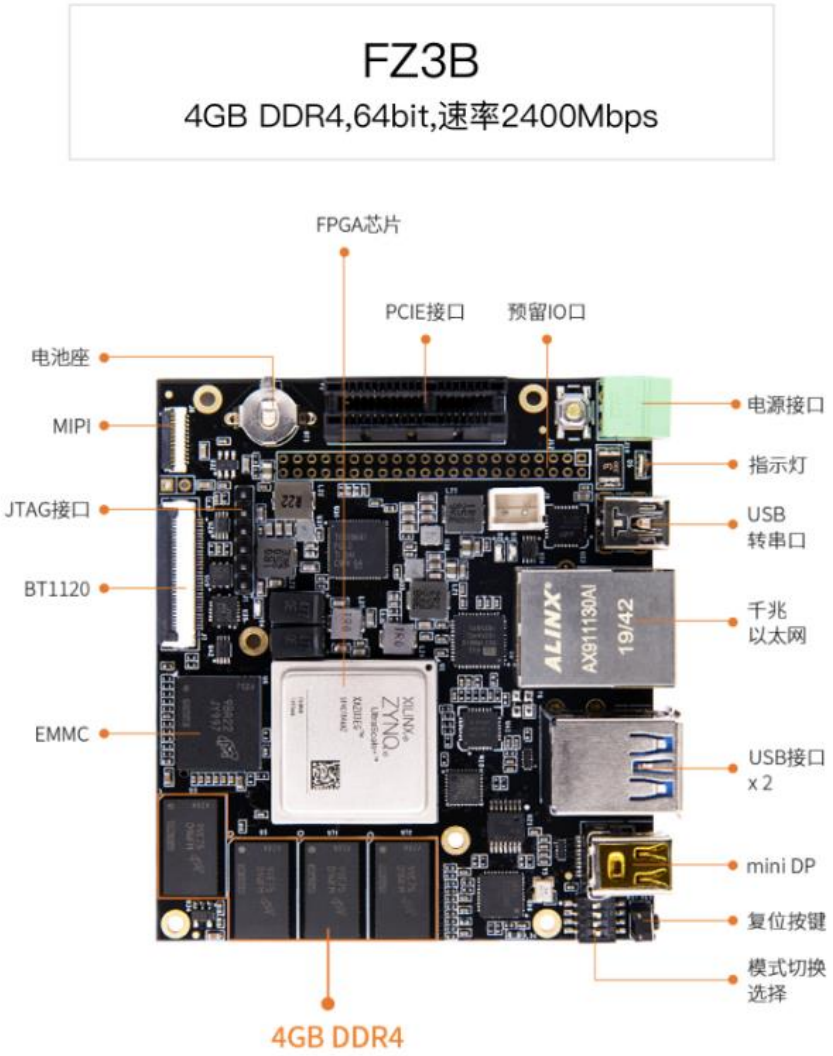


图 8：百度 EdgeBoard 套件简介图

3.2 自行设计的电路板方案及实现

本汽车模型一共装配了三块自行设计的电路板，分别为下位机主控制板、直流电机驱动及电源变换电路板与人机交互电路板，三块电路板通过 FPC 排线连接在一起，构成全功能的下位机系统。

3.2.1 下位机主控制板

下位机主控制板的核心是英飞凌 TC264DA 型芯片，AURIX™ 系列 - TC264DA 系列专用于高级驾驶辅助系统 （ADAS） 领域，适用于雷达和摄像头应用。其具有 200MHz 主频和 DSP 功能的 Dual TriCore™ 计算核心，支持所有内核的浮点和固定点专用 FFT 硬件加速单元，具备高达 2.5MB 的带 ECC 保护的闪存，工作在 125K 周期下的 96KB EEProm，高达 752KB 的 RAM，带 ECC 保护，用于雷达或摄像机的图像存储，带有 4 个 12 位的 SAR AD 转换器，具有 6xSENT，2xPSI5，1xPSI5S 接口、100M 速率以太网接口，1xFlexRay，4xASCLINs，4xQSPI，1xI²C，2xMSC，1xHSSL，I²S emulation，5xCAN 和速率增强型 FD-CAN 接口用于处理器间通信，速率达 2.5Ghz 的 HSSL 接口，用于实时视觉和雷达数据跟踪，采用单电压电源（5V 或 3.3V）易于设计，采用 LQFP-144 封装，免除 X 射线探伤工序，其工作环境温度范围为-40° C 到+125° C。

下位机主控制板具有一些外设接口，如连接 OLED 的 SPI 接口，连接 IMU 的 I²C 接口，输出电机和舵机驱动信号的 PWM 接口以及输入电流电压检测信号的 ADC 接口等。其中无需经常插拔的接口选用 0.5mm 脚距的 FPC 连接器，根据电路板各位置体积要求的不同选用立式或卧式连接器；需要经常插拔的接口则采用 PH2.0 型连接器或 2.54 脚距的排针，在插拔时更加可靠。

3.2.2 直流电机驱动及电源变换电路板

本模型采用分立式 H 桥电路驱动直流电机，MOSFET 型号为无锡新洁能 NCE3050K，MOSFET 驱动器采用 IR2104s，在 MOS 桥臂下侧安装阻值为 0.005Ω 的合金检流电阻，配合 INA199A1 型检流计实现电流检测，进而实现电流闭环控制。用于主控板的 5V 降压电路采用 mp2359 型 Buck 集成芯片，其工作频率为 1.2Mhz，可采用 04020 封装的微型电感，提高电能效率，降低电路体积。用于舵机的 6.5V 降压电路采用 TPS5430DDAR 型 Buck 集成芯片，其工作频

率稍低，仅有 500Khz，但具有 4A 的峰值电流输出能力，且推出时间久，可靠程度高，适合驱动小型伺服舵机。用于 MOSFET 驱动器的 15V 升压电路采用西安航天民芯的 MT3608 型 Boost 集成芯片，其工作频率为 1.2Mhz，平均输出电流为 2A；值得一提的是，这款芯片不能工作于输入电压比设定电压高的状态，否则容易烧毁，这是芯片手册所没有说明的。此外，本团队利用 IR2104s 的 SD 端口对电机驱动器进行使能控制，在不安装大电流开关的情况下便利了设备调试。

3.2.3 人机交互电路板

鉴于比赛规则要求汽车模型安装车壳，不得暴露内部结构，故特别设计了人机交互电路板，安装于车壳外部，包括 OLED 屏幕，四个指示灯，复位按键，三个普通按键，一个五向开关以及一个 2 位拨码开关，便利了人机交互。

4 上位机软件方案及实现

4.1 传统视觉巡线方案

本次第十七届智能车竞赛完全模型组赛道采用的是智能车竞速组别的传统赛道，赛道以蓝色广告布作为背景，黑色胶带和黑色路肩作为赛道边缘，白色 PVC 材质软垫作为赛道。因此我们也在百度 EdgeBoard 板卡上做了传统视觉的巡线算法。百度 Edgeboard 板卡上的系统为针对 Edgeboard 的特殊版本的 ubuntu18.04，可以正常使用 cmake、python-interpreter 等编译器或者解释器，但是对于本次竞速赛来说，我们对比了 python 和 C++的运行效率差异，最终选择了 C++作为本次比赛的项目工程实现语言，通过 cmake 来实现项目代码的编译。

4.2 目标检测方案

4.2.1 图像预处理

对于传统视觉图像处理，我们首先在系统上配置了针对 C++ 的 opencv 库，以用于图像的读取和预处理层面。我们首先把图像缩减成 120*60 的尺寸，在保留下来赛道的基本特征的同时，减少后续图像处理过程中图像的处理数据量，此后利用 opencv 函数将图像处理为灰度图，使用高斯滤波对图像进行模糊，防止噪声对后续的处理产生太大的影响。在使用大津法将灰度图转化为二值化图像，来让图像当中蓝色和黑色的部分都变成黑色，原本为白色的地方还是白色，这样就可以很明显的把赛道特征保留并提取出来，以便于后续的使用。我们还尝试了利用图像腐蚀膨胀的方法进行图像降噪，但若处理缩小后的图像，就较容易过滤掉有用信息。最后转换成二维数组以供后续使用。图像部分预处理代码如下。

```
cv::resize(frame,  frame,Size(Col,Row),  0,  0);

cv::cvtColor(frame,      dst,   cv::COLOR_RGB2GRAY); //转换灰度图  cv::GaussianBlur(dst,
dst,     cv::Size(3,      3),     5,     5);                    //     高     斯     滤     波
cv::threshold(dst,dstt,200,255,    CV_THRESH_BINARY|THRESH_OTSU);//二值化图像    |
    THRESH_OTSU
//   element   =   getStructuringElement  (MORPH_RECT ,    Size(3,    3));   //采用腐蚀膨胀降
噪

Get_Pixle(dstt);        //转换为二维黑白数
```

图 9：图像部分预处理代码图

4.2.2 赛道中线和偏差提取

在二值化之后的图像仅有黑白两个色块，因此赛道边缘的特征十分的明显，即在边缘处存在黑白跳变，这样我们就可以把赛道边缘提取出来。本次比赛我们为了减小运算量而没有采取遍历整张图片的方式，而是利用边线是连续的的特征，可以根据上一行的边界所在的位置近似确定下一行边线的位置，这样可以快速找到两条边线。再用两条边线拟合出中线，通过中线求出目前小车偏离中线的误差，再以此误差作为舵机 PID 控制器的输入，以此控制小车的姿态。搜线的部分代码放在附录。中线提取效果如图 10 所示。

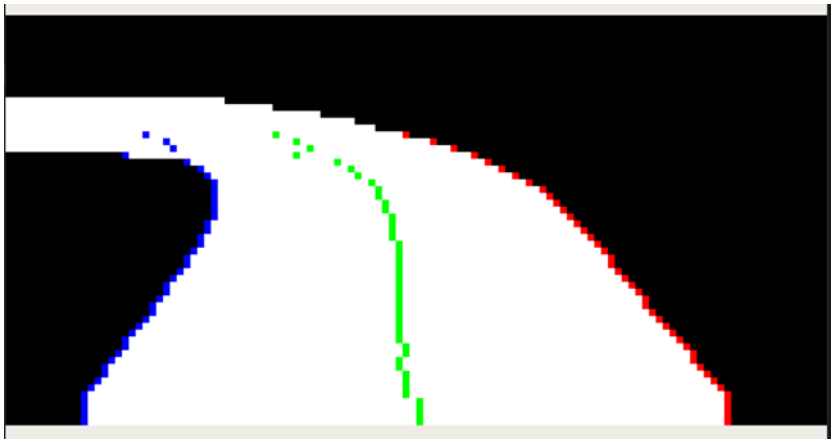


图 10：中线提取效果图

4.2.3 目标检测

由于本次完全模型组当中有一些任务点的标志存在，因此需要模型汽车上位机具有能够使用深度学习目标检测的功能。例如泛行区、施工区、坡道、禁行标志以及我们对与车库标志的识别，都采用了使用深度学习目标检测识别的方法。对于此次目标检测任务，我们采用的方案是另开一个子线程，巡线任务作为父线程，子线程使用父线程的图像数据资源，父线程使用子线程的推理结果。为了让深度学习推理子线程不占用太多的 CPU 资源，

所以我们尽可能选择较为轻量级的网络，同时要保持准确率。我们首先尝试了 yolov3-mobilenet-v1，但是训练好模型之后，不管输入图像的尺寸是较小的 320*320 还是较大的 608*608，在两个线程共同运行的时候会感觉到很严重的体感延迟，导致舵机相应过慢，效果很差。于是我们选择了更为轻量级的 ssd-mobilenet-v1,让我们能将模型预测置信度较高的情况下保持一个较高的预测速度，同时最后模型检测到目标的位置（检测框）也能相对保持在一个稳定的范围之内。目标检测效果图如图 11 所示。

图 11：目标检测效果图

5 下位机软件方案及实现

软件是小车的灵魂，一辆调试完美的智能车通过微调参数应该能够适应不同环境的赛道，并且在不同环境的赛道上跑出理想的速度，它是由可靠的硬件电路、合理的机械调整、稳定的软件算法来实现，而软件算法的稳定性是至关重要的。

由于今年全模型组的车较重，所以在软件控制中不仅仅要考虑车子的稳定性，还要考虑耗能，平滑的控制不仅仅可以让车子更加稳定，而且会更加节能，所以软件控制主要是让信标车在整个过程中保持平滑的控制，速度环、方向环控制思路和参数就显得尤为重要，再者由于用的 TC264 单片机是双核处理器，在软件算法上要做出优化。

5.1 主程序结构

车模运行程序的主体结构有以下四个部分：

1.底层初始化

2.串口，陀螺仪，编码器数据采集

3.元素判断

4.方向，速度控制

5.1.1 底层初始化

对 TC264 单片机做基本的初始化，包括对时钟、GPIO 口、计时器、串口、SPI、ADC、FLASH、PID、OLED、编码器、陀螺仪等初始化。(具体实现略)

5.1.2 参数设定

我们在人机交互小板上设置了按键，用于智能车选择档位，在 OLED 上显示菜单参数，以在比赛时对车作适当地调整。

5.1.3 传感器数据采集处理

通过上位机发送串口数据接收判断当前情况，控制车辆行进方向和车速。

5.2 伺服及驱动器件控制算法

5.2.1 行驶算法实现

智能车通过摄像头和 IMU 测量当前状态。循迹算法可以根据位置状态对舵机角度，电机速度等控制量进行调整以保证智能车沿线正常行驶。具体计算方法为路径规划定理。控制小车行驶。结合模糊 PID 控制器，可使小车稳定巡线。并确定距离。

5.2.2 方向控制算法

方向的控制是根据传感器检测到的图像来实现的，关于摄像头的个数和排布，不同的控制方法有不同的选择，摄像头个数减少会带来处理上的方便，但也会使采集到的信息不够丰富，摄像头个数增加会使得信息处理稍显复杂，但是到的赛道信息较丰富。我们权衡后选择两个摄像头来进行处理。方向的控制基本分为方向判断和速度大小。

转向判断：我们利用上位机判断的位置信息，控制舵机转向角度。

速度控制：根据对调试情况设定车辆速度，得出模糊量后，对模糊量进行 PD 控制，通过修改模糊规则查询表和 PD 项的参数能够使得小车在赛道上实现较好的速度控制。

5.2.3 速度控制算法

PID 控制算法通过调整比例、积分和微分三项参数实现在反馈调节机制，以其技术成熟、易被人们熟悉和掌握、不需要建立数学模型、控制效果好、鲁棒性等优点而广泛应用于控制系统。

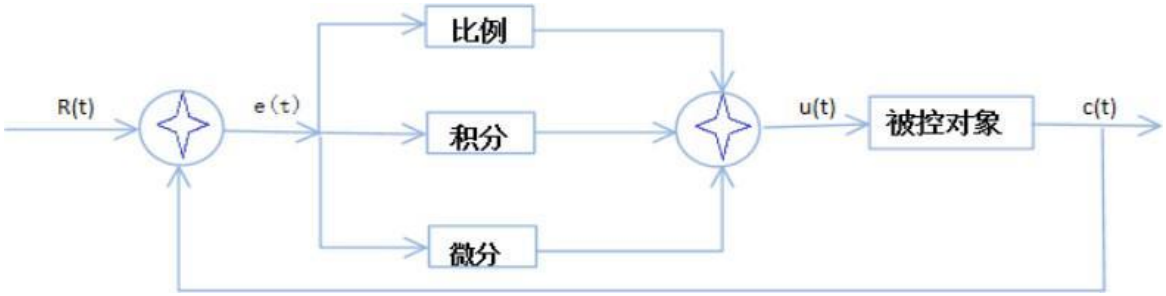


图 11：PID 算法流程图

PID 算法依其具体实现可分为位置式 PID 和增量式 PID。为了降低计算量及得到稳定的结果，我们使用了增量式 PID，故以下主要对增量式 PID 介绍。

$$u(t) = K_p\left[e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d\frac{de(t)}{dt}\right]$$

（公式 1）

公式 1 中 e(t)为系统偏差,e(t)=r(t)-c(t);Kp 为比例系数;Ti 为积分时间常数;Td 为微分时间常数。

实际计算时，需将上式离散化，微分用差分代替，积分用和式代替，如下式：

$$u(k)=K_p*e(k)+K_i*\sum_{j=0}^{k}e(j)+K_d*[e(k)-e(k-1)]$$

（公式 2）

公式 2 可写成：

$$u(k) = K_P\{e(k) + \frac{T}{T_I}\sum_{j=0}^{k}e(j) + \frac{T_D}{T}[e(k) - e(k-1)]\}$$

（公式 3）

其中，Ki=Kp*T/Ti,Kd=Kp*Td/T，可继续推导：

$$u(k-1)=K_P\{e(k-1)+\frac{T}{T_I}\sum_{j=0}^{k-1}e(j)+\frac{T_D}{T}[e(k-1)-e(k-2)]\}$$

$$\Delta u(k)=K_P\{[e(k)-e(k-1)]+\frac{T}{T_I}e(k)+\frac{T_D}{T}[e(k)-2e(k-1)+e(k-2)]\}$$

$$=K_P\Delta e(k)+K_I e(k)+K_D[\Delta e(k)-\Delta e(k-1)]\}$$ （公式 4）

式中 $\Delta e(k)=e(k)-e(k-1)$； $K_I=K_P\frac{T}{T_I}$； $K_D=K_P\frac{T_D}{T}$

由此可以看出，如果计算机控制系统采用恒定的采样周期 T，一旦确定 Kp、Ki、Kd，只要使用前后三次测量的偏差值，就可以求出控制量 Kp、Ki、Kd。

PID 控制器参数整定的方法很多，例如试凑法、临界比例度法、扩充临界比例度法等。我们一般选择相对操作简单的试凑法。

一般试凑法确定参数的步骤如下：

1.确定比例系数 Kp

确定比例系数 Kp 时，首先去掉 PID 的积分项和微分项，可以令 Ti=0、Td=0， 使之成为纯比例调节。输入设定为系统允许输出最大值的 60%～70%， 比例系数 Kp 由 0 开始逐渐增大，直至系统出现振荡；再反过来，从此时的比例系数 Kp 逐渐减小，直至系统振荡消失。记录此时的比例系数 Kp，设定 PID 的比例系数 Kp 为当前值的 60%～70%。

2.确定积分时间常数 Ti

比例系数 Kp 确定之后，设定一个较大的积分时间常数 Ti，然后逐渐减小 Ti， 直至系统出现振荡，然后再反过来，逐渐增大 Ti，直至系统振荡消失。记录此时的 Ti，设定 PID 的积分时间常数 Ti 为当前值的 150%～180%。

3.确定微分时间常数 Td

微分时间常数 Td 一般不用设定，为 0 即可，此时 PID 调节转换为 PI 调节。如果需要设定，则与确定 Kp 的方法相同，取不振荡时其值的 30%。

系统空载、带载联调对 PID 参数进行微调，直到满足性能要求。

全模型组车对控制平滑度要求很高，所以对控制算法有更高的要求，在保持车子稳定和节能的上提高速度，就要通过不懈努力，多加尝试，在一个好的算法基础上，找到最优的参数，从而使车子跑出理想成绩。当然也要不断创新， 不拘于当前算法，大胆尝试。

6 总结

6.1 创新点

创新点 1：本辆智能汽车使用“传统视觉+深度学习”的方法，通过计算机视觉相关领域的新老两种方法结合，实现了对智能汽车对于车道线、目标的识别，并在识别后执行相应的动作控制算法，以达到任务要求。

创新点 2：本辆智能汽车对于各种目标识别算法进行了测试优化，同时对用于车道线回归的神经网络也进行了优化，使得我们团队的模型更加适合于较多的环境，增强了模型的泛化能力，使得完成任务的质量和稳定性有所提升。

创新点 3：本辆智能汽车使用多线程同时运行多个任务，让传统巡线、目标检测、上下位机通讯等问题分开处理，同时保证线程间通信的安全性、准确性，同时提高了上位机的运算效率。

6.2 技术路线

在底层上面，底层的主要使用了英飞凌 TC264 作为底层控制芯片，来控制小车的舵机转向、电机转动。巡线方面，我们使用了传统视觉进行巡线的算法，来达到快速、准确的巡线效果，让小车在较快运行的时候也能准确无误。目标检测方面，我们使用了 ssd-mobilenet-v1，这是一个通过数据优化并减少了输入数据量大小的的卷积神经网络。能够轻松地搭载在小车上。由于其所用到的算子较少，也能够适配小车系统方面的算子限制。硬件方面，我们使用了一块主控、一块驱动两块板子，使用串口完成上下位机通讯，让上位机发出的指令下位机能够做到快速响应，让小车能够迅速做出姿态调整。

6.3 心得体会

我们的车模结合了深度学习，计算机视觉，运动控制等相关学科和知识，很好的应用了当下最流行的人工智能、自动驾驶领域的相关知识和课题，最终我们团队的智能车能够很好的自主完成车道线巡航及各种目标识别和各种任务的完成。

在百度完全模型组的任务中，我们传统视觉巡线算法推算出当前车道线中线跟车身中线的偏差，并且进行校正，使得智能车在车道线之间稳定运行。我们结合了百度 Paddle Paddle 深度学习的框架和基于 Paddle Mobile 全新升级推出的 PaddleLite 的端侧推理引擎，使得车模在场地内进行实时画面的读取，并基于 ssd-mobilenet-v1 的目标检测模型来进行对于标志物的识别，通过对不同标志物采取不同的处理方式， 通过给下位机不同的指令，从而完成比赛任务。

本次比赛也是对我们作为自动化学院的学生综合实力的一种检验，是我们平时课内学习到的知识的一种实践，同时作为新时代青年，努力学习应用层面的知识，深入探究其中原理，是我们应该做的。平时在竞赛中，我们也接触各种集成电路、芯片、上位机、控制算法、图像处理算法等，都是课上讲过的或者接触过的一些知识，而这种比赛就是我们实践的机会。在这之中，我们应该认识到我们平时的学习在实践当中的重要性，因此更应该为了成为国内高级技术人才而努力奋斗。同时，这次备赛更加锻炼了我团队协作的能力，让我们每个分工不同的人紧紧联系在一起，让整个流程得以实现，让我也意识到一个团结的团队是多么重要，让我感受到了大家的力量。

7 致谢

在本次智能车国赛备赛期间，我们一路披荆斩棘。从方案设计、芯片选型，到后来的原型建造与迭代。问题不断浮现，也不断得到解决，在此过程中，我们发现，在技术上我们不断成长，思想上不断成熟。在这过程中，离不开学校、老师和同学的支持。首先，感谢学校对这次比赛的重视，感谢学校教务处对我们备赛的大力支持。其次，我们要感谢余善恩和黄继业老师的悉心教导， 感谢两位老师在我们遇到困难的时候的鼓励与帮助，没有他们在思想上的指导和整体的规划安排，我们绝不会有今天的成果；最后，我们要感谢实验室的同志们，感谢大家长期以来的鼓励和陪伴，感恩我们同时工作在这么和谐友爱的实验室里，感恩一个个不眠之夜里不辞辛劳的身影，感谢为比赛默默付出的兄弟姐妹，这段智能车之旅是我们每个人都难以忘怀的大学记忆。

8 参考文献

[1]邵贝贝，《单片机嵌入式应用的在线开发方法》，北京清华大学出版社 2004 年 10 月第 1 版

[2]卓晴，黄开胜，邵贝贝 《学做智能车》 北京-北京航空航天大学出版社 2007

[3]谭浩强 《C++程序设计》 北京-清华大学出版社 2004

[4]潘松，黄继业 《现代数字电路基础教程》 北京-科学出版社 2008

[5]王水平 《开关稳压电源原理及设计》 北京-人民邮电出版社 2008

[6]TI 公司 Use of Rail-to-Rail Operational Amplifiers/Application Report 1999

[7]英飞凌公司 BTS7970 Rev2.0/ Datasheet 2006

[8]钱江一号第六届“恩智浦”杯全国大学生智能车大赛技术报告杭州电子科技大学 2011

[9]杭电节能一队第十四届“恩智浦”杯全国大学生智能车大赛技术报告，杭州电子科技大学，2019

[10]第五届“恩智浦”杯全国大学生智能车大赛清华大学三角洲电磁队技术报告

[11]韩绍坤，许向阳，王晓华编《自动控制原理》北京理工大学出版社，2009

[12]王盼宝主编《智能车制作 从元器件、机电系统、控制算法到完整的智能车设计》，清华大学出版社

[13]杭电节能信标一队第十六届“恩智浦”杯全国大学生智能车大赛技术报告，杭州电子科技大学，2020

9 附录A部分程序源代码

```
/*
 *  MainControl.c
 *  Created by 章生瀚 on 2021/1/29
 *  All rights reserved.
 */

#include "headfile.h"
#include <math.h>
#include <stdlib.h>

uint8 PID_flag = 0;
uint32 RUN_MODE = 4;    //0:正常运行    1:测试模式  2:充电模式  4：未发车

int BT_FLAG = 0; //0:不启动蓝牙调试   1：启动蓝牙调试
uint8 sing_flag = 0; //娱乐项目 放歌玩
void allInit (void)
{
    get_clk();
//    cmd_init();

    gpio_init(KEY1_PIN, GPI, 1, PULLUP);
    sing_flag = gpio_get(KEY1_PIN);
    if (sing_flag)
    {
        beepInit();
        beep_status = BEEP_ENABLE;
    }
    else
    {
        gtm_pwm_init(ATOM1_CH5_P00_6, 3000, 0);
        beep_status = BEEP_SING;
    }
```

```
//    beepInit();
    Serial1.init();
    oled_init();
    oled_p6x8str(22, 1, "Ubuntu 20.04");
    oled_p6x8str(0, 2, "[OK]Oled connected");
    cmd_send("[OK]Oled connected\n");
    //        systick_delay_ms(STM0,50);
    flash_read_from_eeprom(); //选择读取数据
//    oled_p6x8str(0, 3, "waiting SD card...  ");
//    while(1){
//        int res = SD_init();
//        oled_p6x8str(0, 3, "waiting SD card...  ");
//        switch(res){
//            case 1:oled_p6x8str(0, 4, "1");break;
//            case 2:oled_p6x8str(0, 4, "2");break;
//            default:oled_p6x8str(0, 4, "ok");
//        }
//        if(res == 0) break;
//    }
    oled_p6x8str(0, 3, "waiting IMU...      ");
    cmd_send("waiting IMU...      \n");
    zf_mpu6050_init();
//    while(mpu_dmp_init());
    oled_p6x8str(0, 3, "[OK]IMU connected   ");
    cmd_send("[OK]IMU connected\n");
    //        systick_delay_ms(STM0,50);

    oled_p6x8str(0, 4, "waiting Encoder...  ");
    cmd_send("[OK]Oled connected\n");
    encoderInit(&main_motor, ENCODER_CHANNEL, ENCODER_COUNT_PIN, ENCODER_DIR_PIN);
    oled_p6x8str(0, 4, "[OK]Encoder connected");
    cmd_send("[OK]Encoder connected\n");

    oled_p6x8str(0, 5, "waiting Motor...  ");
    motorInit(&main_motor, MOTOR_PIN_1, MOTOR_PIN_2, 9 * 1000, I_LOOP, 6000, 1); //初始化
电机 ，把电机引脚连接。
    ISenseInit();
    pidMotorInit(&main_motor); //初始化电机 PID
    pidMotorSet_V_Para(&main_motor, Motor_Kp, Motor_Ki, Motor_Kd); //设定 PID 速度环参数
    pidMotorSet_I_Para(&main_motor, I_Kp, I_Ki); //设定 PID 电流环参数
```

```
    oled_p6x8str(0, 5, "[OK]Motor connected");
    cmd_send("[OK]Motor connected\n");

    oled_p6x8str(0, 6, "waiting Button...  ");
    buttonInit();
    sMenuInit();
    menu_pointer = &PID_menu[0]; //初始化菜单
    cmd_send("[OK]menu connected\n");
    oled_p6x8str(0, 6, "[OK]Button connected");
    cmd_send("[OK]Button connected\n");

    BT_init();  //蓝牙初始化
    Vol_init(); //电压检测初始化
    ServoInit(); //舵机初始化

//    flash_read_from_eeprom();//选择读取数据
    cmd_send("[OK]eeprom connected\n");

    oled_fill(0x00);
//    adc_init(ADC_0,ADC0_CH0_A0);//电流检测

    pit_interrupt_ms(CCU6_0, PIT_CH0, 20); //打开定时器中断
    pit_interrupt_ms(CCU6_0, PIT_CH1, 5); //打开定时器中断
//    if(main_motor.motor_control_mode == I_LOOP)
    pit_interrupt_ms(CCU6_1, PIT_CH0, 5); //ADC 电流环采集运算

    cmd_send("[OK]Timer running\n");
    PID_PositionInit(&main_servo);
    pidPosSet(&main_servo, Pos_Kp, Pos_Ki, Pos_Kd);
    main_servo.set = 128;
    cmd_send("[OK]PID inited\n");
    IfxCpu_emitEvent(&g_cpuSyncEvent);
    IfxCpu_waitEvent(&g_cpuSyncEvent, 0xFFFF);

    enableInterrupts();
//    main_motor.I_set_value = 800;
}
```

int run_flag = FALSE; //用来控制程序运行状态 0 为未向上位机发送开始指令，不接受上位机的指令 1 为已发送开始，可以接受上位机发来的数据

```
uint32 SetSpeed = 0; //实际设定速度（对这个值赋值会改变当前电机速度）
uint32 SetCurveSpeed = 170; //弯道速度
uint32 SetStraightSpeed = 260; //直道速度
uint32 change_angle = 5; //角度偏差 判断直道还是弯道速度
uint32 down_speed_Kp = 100; //减速 Kp
/*************************************************/
uint32 sc_speed = 200;
uint32 sg_speed = 200;
uint32 ring_speed = 200;
uint32 jy_speed = 200;
/*************************************************/
uint32 out_park_speed = 0; //出库速度
uint32 out_park_angle = 0; //出库速度
uint32 out_park_time = 0; //出库时间

uint32 park_speed = 0; //入库速度
uint32 park_angle = 0; //入库速度
uint32 park_time = 0; //入库时间
/*************************************************/
//主函数控制
void mainControl (void)
{
    updateFromPC(); //从串口更新数据
    Vol_get(); //检测电压
    if (low_battary_flag == LOW_POWER) //如果低电压则不运行
    { //低电压返回状态
        main_servo.inter = 0;
        main_servo.set = 128;
        main_motor.set_speed = 0;
        for (int i = 5; i > 0; i--)
        {
            beepStatus(BEEP_ON);
            systick_delay_ms(STM0, i * 100);
            beepStatus(BEEP_OFF);
            systick_delay_ms(STM0, i * 100);
        }
        while (1)
        {
            beepStatus(BEEP_ON);
            systick_delay_ms(STM0, 1 * 100);
```

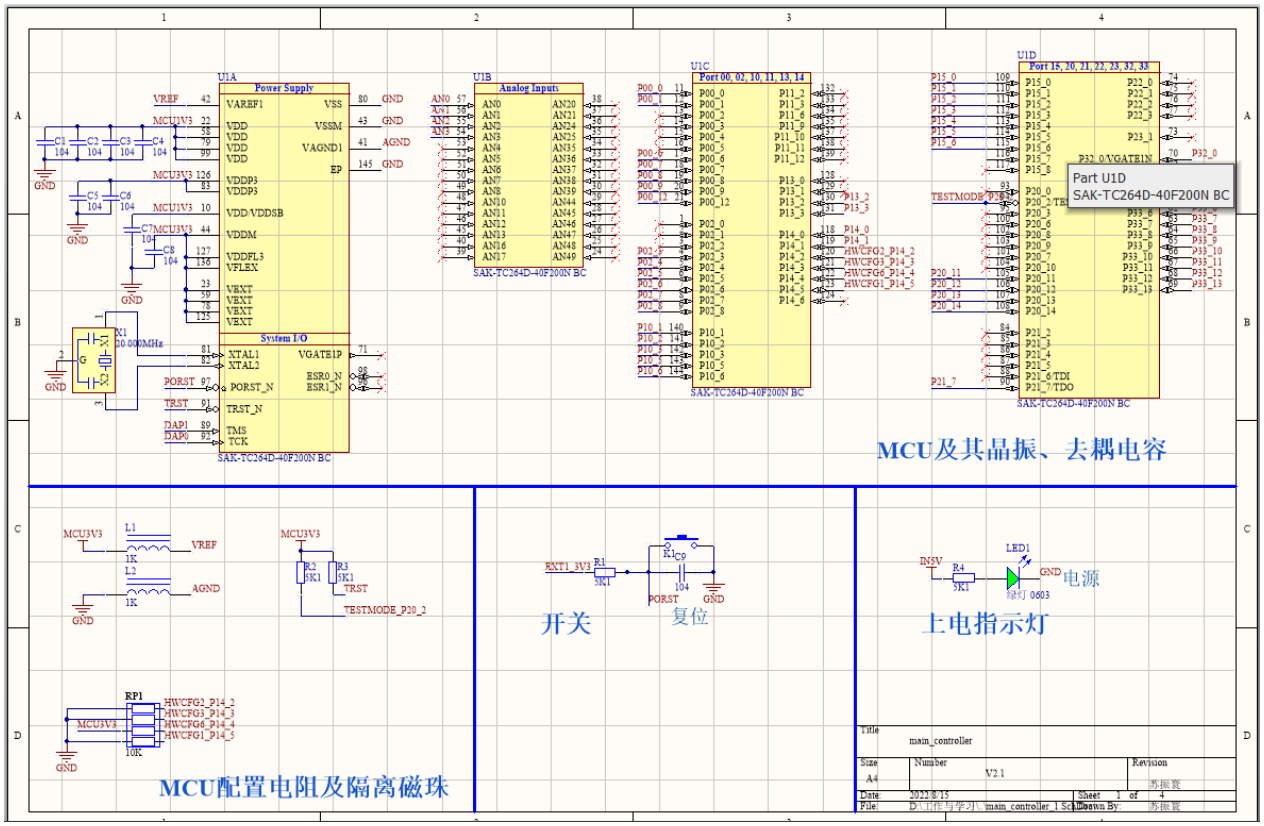
```
            beepStatus(BEEP_OFF);
            systick_delay_ms(STM0, 1 * 100);
        }
        return;
    }
    if (run_flag == RUN_START) //开始运行
    { //判断是否为运动状态
        updateFromPC();
        if (Serial1.take_byte == TAKE_NO)
        {
            pidPosSet(&main_servo, Pos_Kp, Pos_Ki, Pos_Kd);
            beepStatus(BEEP_OFF);
            main_servo.set = Serial1.mid_err;
            main_motor.set_speed = SetStraightSpeed
                    - abs(SetStraightSpeed - SetCurveSpeed) / 128.0 * abs(main_servo.set
- 128); //速度控制
        }
        else if (Serial1.take_byte == TAKE_ALL) //由上位机接管所有速度
        {
            beepStatus(BEEP_OFF);
            pidPosSet(&main_servo, Pos_Kp, Pos_Ki, Pos_Kd);
            main_servo.set = Serial1.mid_err;
            main_motor.set_speed = Serial1.take_speed;
        }
        else if (Serial1.take_byte == TAKE_ELE)
        {
            if (Serial1.flag_byte == ELE_LEFT_RING || Serial1.flag_byte ==
ELE_RIGHT_RING)
            {
                pidPosSet(&main_servo, ring_Pos_Kp, ring_Pos_Ki, ring_Pos_Kd); //如果是圆
环，则重新设置转向PID
                main_motor.set_speed = ring_speed;
            }
            else if (Serial1.ai_ele == AI_ELE_BRANCH)
            { //三叉
                pidPosSet(&main_servo, Pos_Kp, Pos_Ki, Pos_Kd); //如果是圆环，则重新设置
转向PID
                main_motor.set_speed = sc_speed;
            }
            else if (Serial1.ai_ele == AI_ELE_CONSTRUCTION)
```

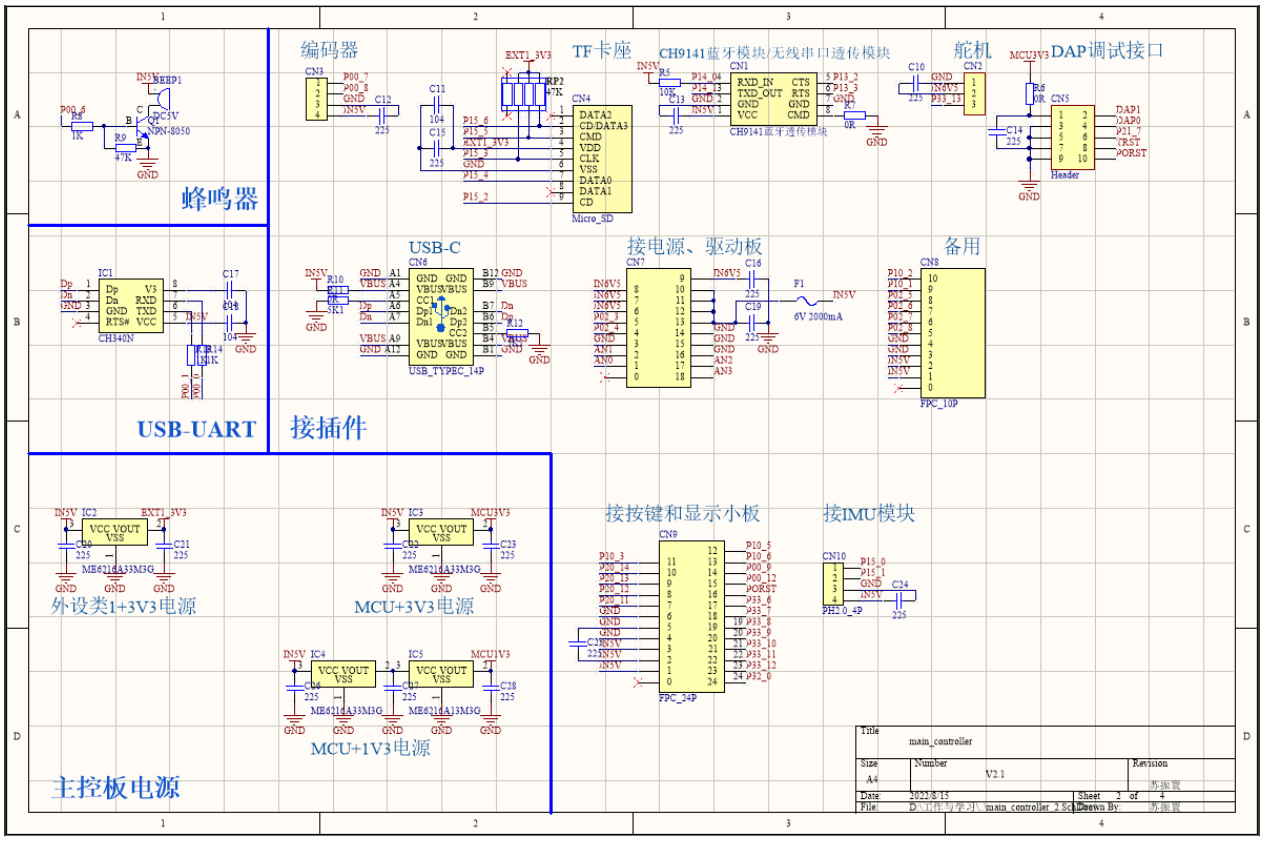
```
            { //施工区
                pidPosSet(&main_servo, Pos_Kp, Pos_Ki, Pos_Kd); //如果是圆环，则重新设置
转向PID
                main_motor.set_speed = sg_speed;
            }
            else if (Serial1.flag_byte == ELE_PARK || Serial1.ai_ele == ELE_PARK)
            {
                pidPosSet(&main_servo, Pos_Kp, Pos_Ki, Pos_Kd); //如果是圆环，则重新设置
转向PID
                main_motor.set_speed = park_speed;
                run_flag = PARKING;//设置停车
            }
            else if (Serial1.ai_ele == AI_ELE_GASSTATION)
            { //施工区
                pidPosSet(&main_servo, Pos_Kp, Pos_Ki, Pos_Kd); //如果是圆环，则重新设置
转向PID
                main_motor.set_speed = jy_speed;
            }
            main_servo.set = Serial1.mid_err; //舵机角度
            beepStatus(BEEP_ON);
        }
    }
    else if (run_flag == RUN_STOP) //未开始运行
    {
        updateFromPC();
        main_servo.inter = 0;
        main_servo.set = 128; //舵机角度
        main_motor.set_speed = 0;
    }
    else if (run_flag == READY)
    {
        main_servo.set = out_park_angle; //舵机角度
        main_motor.set_speed = out_park_speed;
        systick_delay_ms(STM0, out_park_time);
        run_flag = RUN_START; //发车完成
        for (int i = 0; i < 3; i++)
        {
            uart_putchar(CMD_UART, 0xFF);
            uart_putchar(CMD_UART, 0x00);
            uart_putchar(CMD_UART, 0xCC);
```

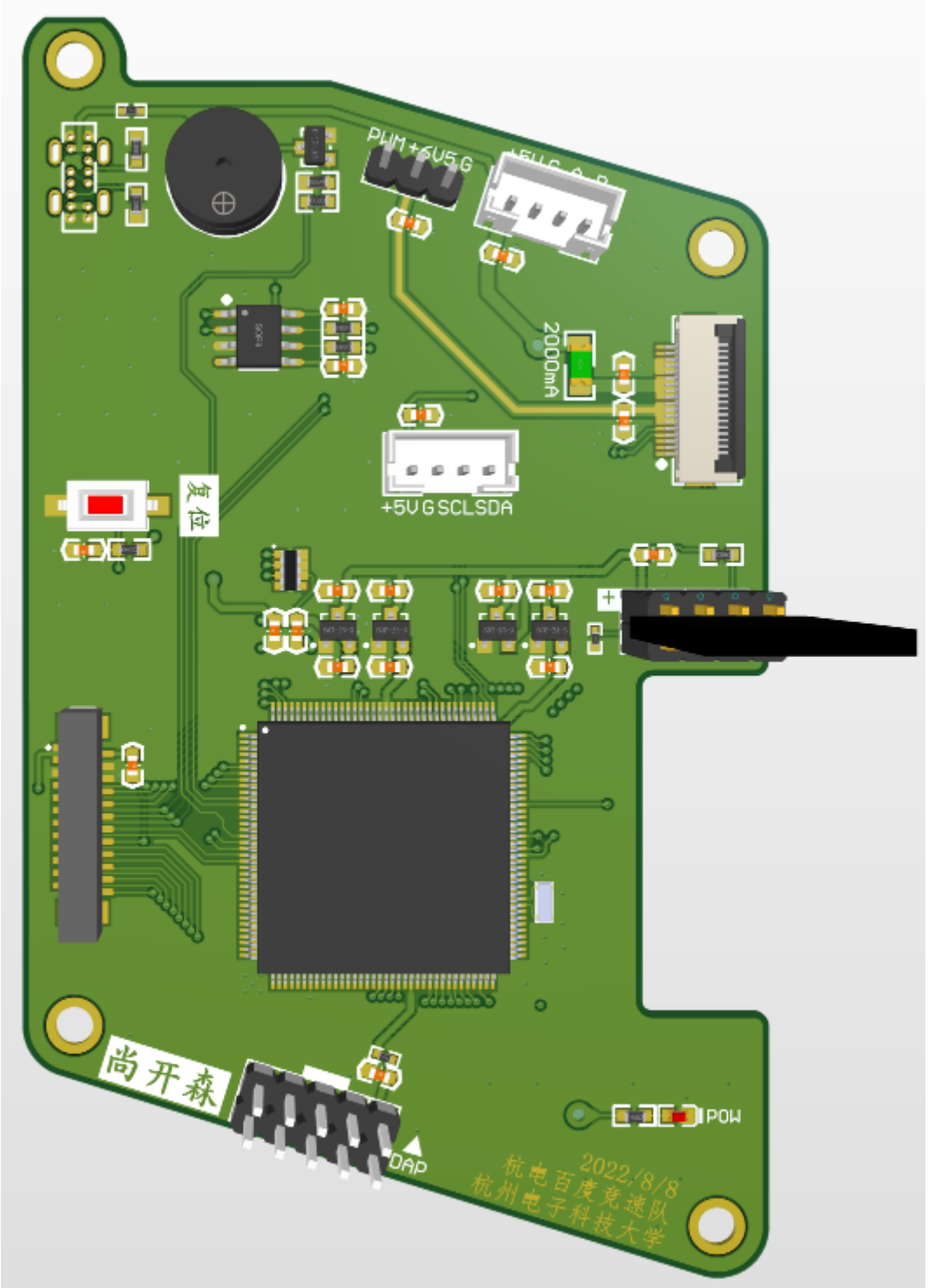
```
            systick_delay_ms(STM0, 10);
        } //发十次
    }
    else if (run_flag == PARKING)
    {
        main_servo.set = park_angle; //舵机角度
        main_motor.set_speed = park_speed;
        systick_delay_ms(STM0, park_time);
        run_flag = RUN_STOP; //入库完成
    }
}
```

10 附录B系统原理及模型图

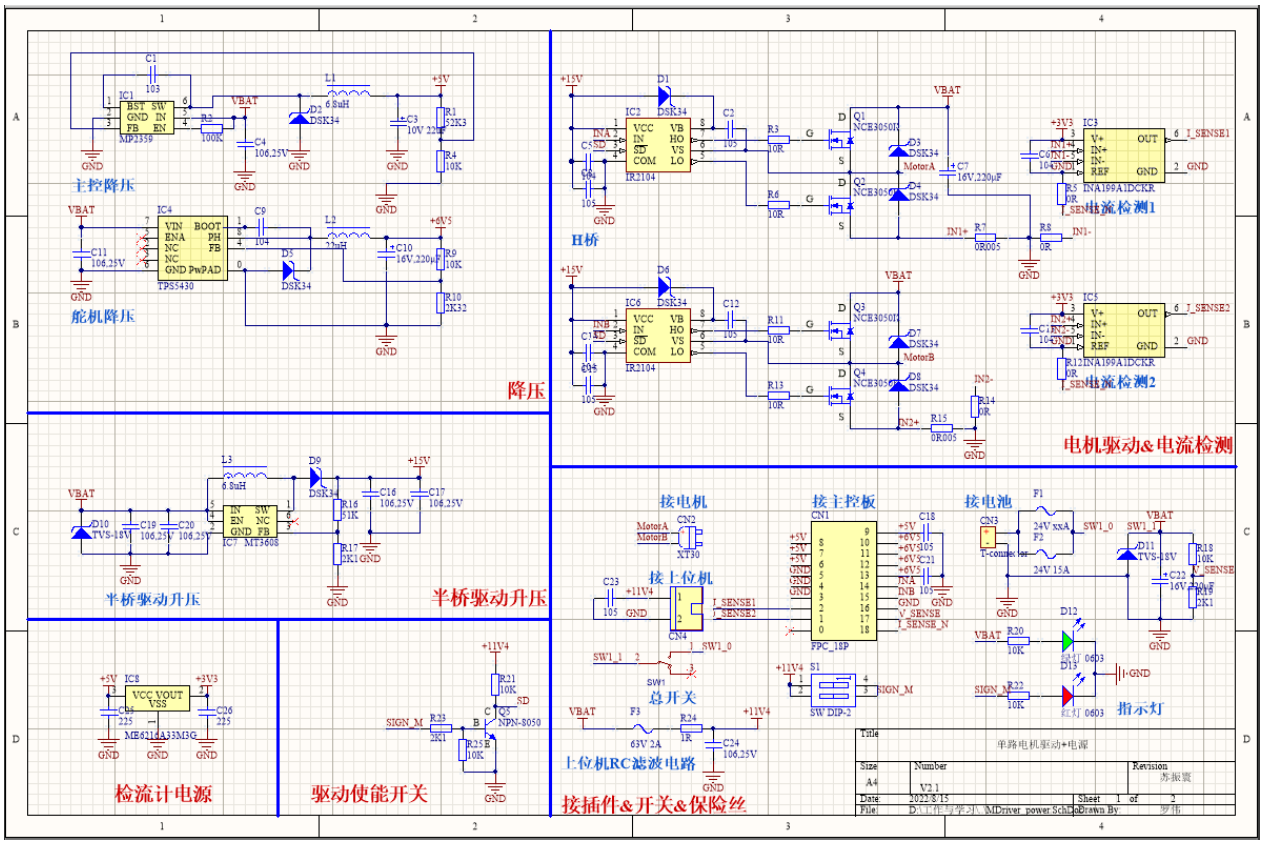
10.1 下位机主控制板：

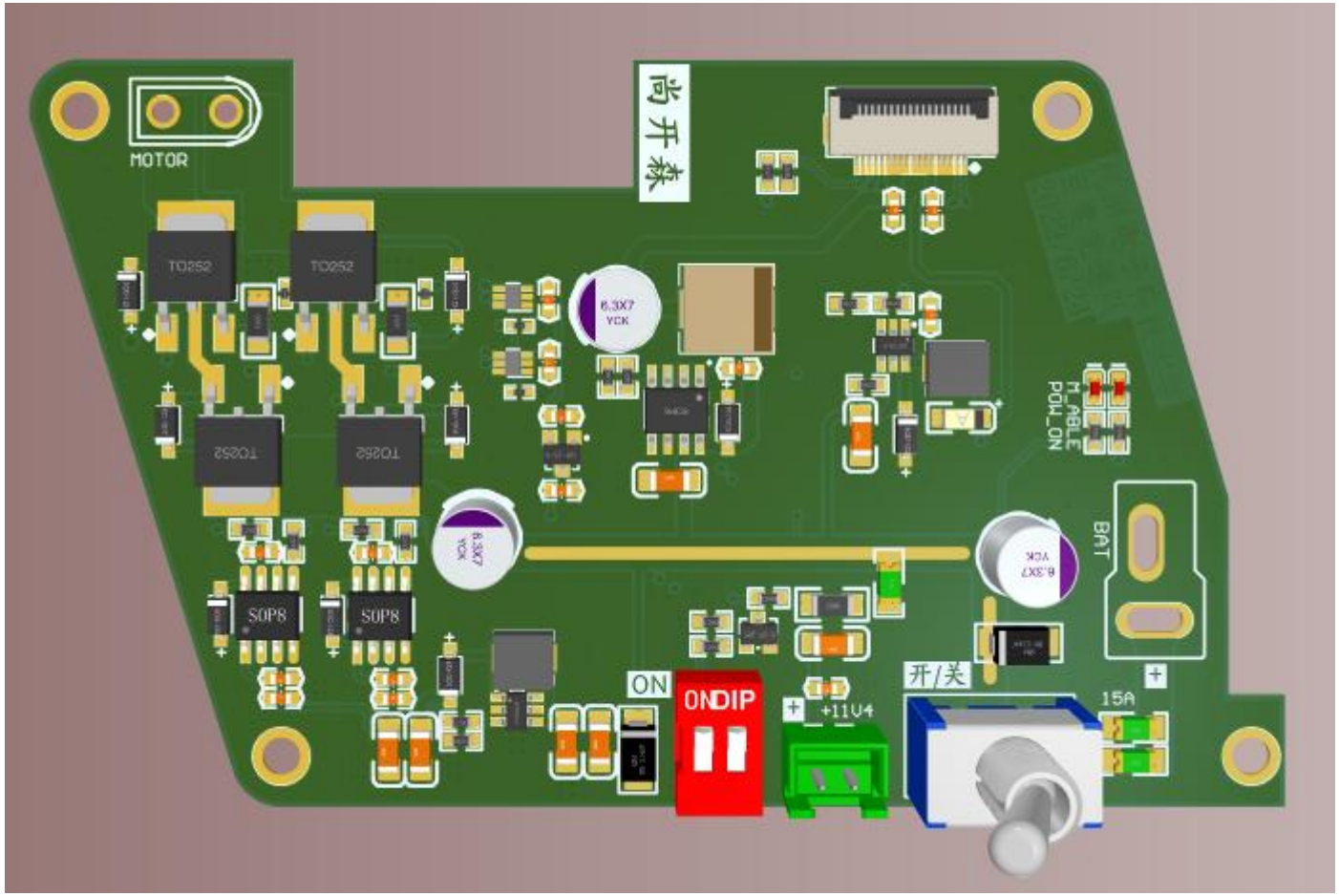



蜂鸣器
编码器
TF卡座
CH9141蓝牙模块/无线串口透传模块
舵机
DAP调试接口
USB-UART
USB-C
接插件
接电源、驱动板
备用
外设类1+3V3电源
MCU+3V3电源
接按键和显示小板
接IMU模块
MCU+1V3电源
主控板电源
Micro_SD
USB_TYPEC_14P
FPC_10P
FPC_24P
CH340N
ME6216A33M3G
Header
6V 2000mA
Title
main_controller
Size
A4
Number
V2.1
Revision
Date
2022/8/15
Sheet
2
of
4
File
D:\工作与学习\main_controller_2.SchDoc
Drawn By
苏振寰

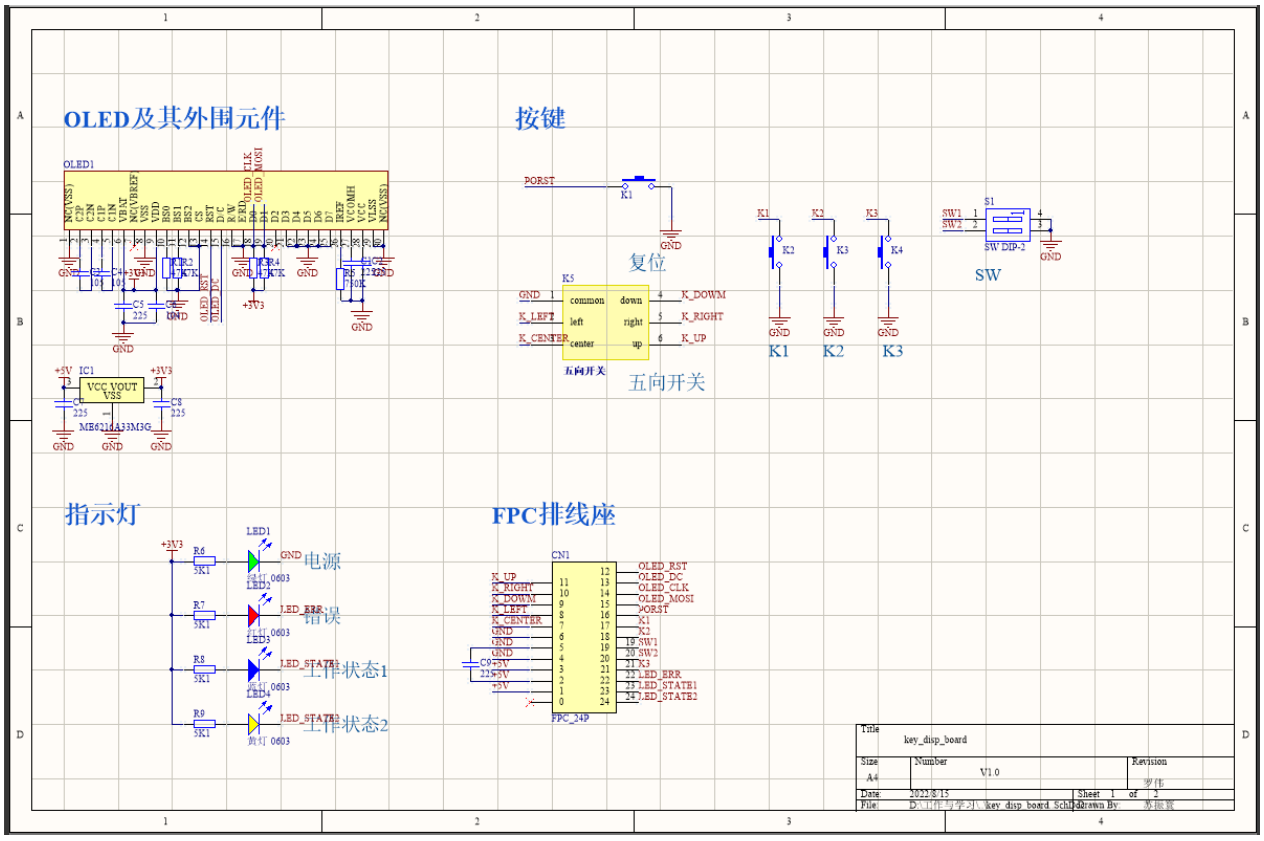

PWM+5V5G
2000mA
+5V G SCL SDA
复位
尚开森
DAP
POW
2022/8/8
杭电百度竞速队
杭州电子科技大学

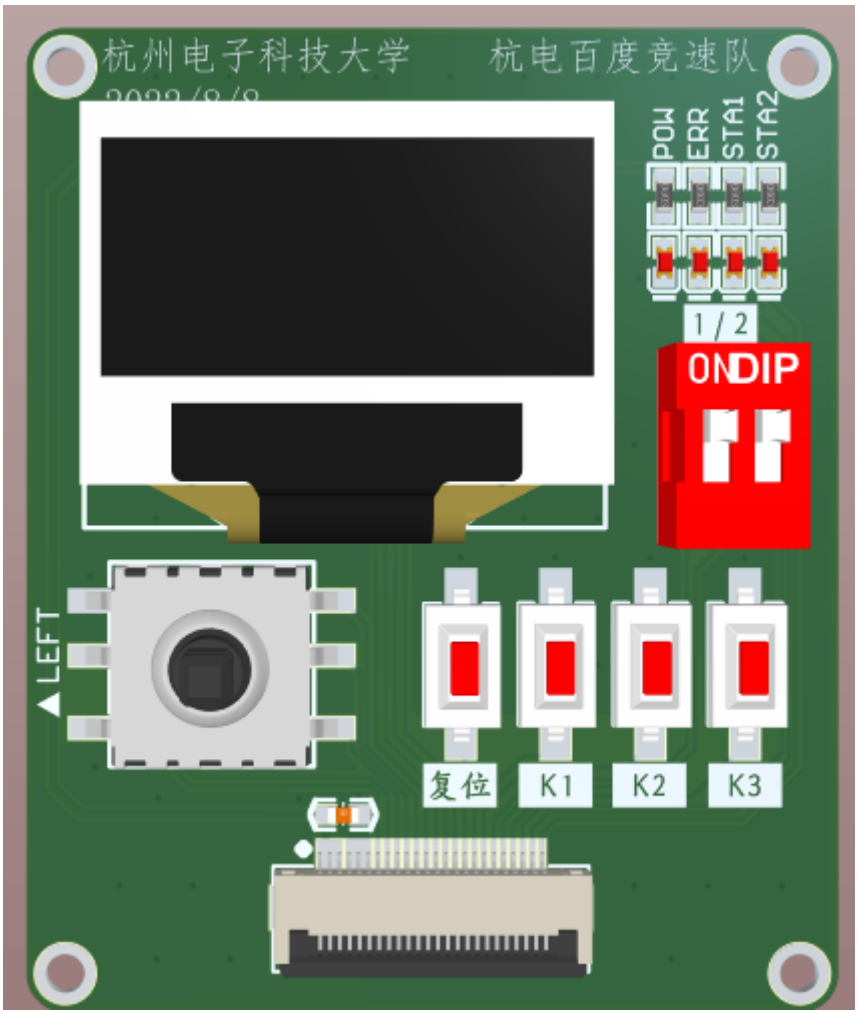
10.2　直流电机驱动及电源变换电路板：




MOTOR
尚开森
TO252
TO252
TO252
TO252
6.3X7
YCK
SOP8
SOP8
ON
ONDIP
+11V4
开/关
15A
BAT
H_ABLE
POW_ON

10.3　人机交互电路板：


OLED及其外围元件
按键
指示灯
FPC排线座
复位
五向开关
SW
电源
错误
工作状态1
工作状态2
key_disp_board
V1.0
2022/8/15


杭州电子科技大学
杭电百度竞速队
POW
ERR
STA1
STA2
1 / 2
ON
DIP
LEFT
复位
K1
K2
K3