第十七届全国大学生

智能汽车竞赛

技 术 报 告

中国矿业大学（北京）

CHINA UNIVERSITY OF MINING AND TECHNOLOGY-BEIJING

学　　校：中国矿业大学（北京）

队伍名称：地灵殿的装修队

参赛队员：邹放达

魏泽洋

林锐东

指导教师：赵锋　查雯婷

关于技术报告和研究论文使用授权的说明

本人完全了解全国大学生智能汽车竞赛有关保留、使用技术报告和研究论文的规定，即：参赛作品著作权归参赛者本人，比赛组委会可以在相关主页上收录并公开参赛作品的设计方案、技术报告以及参赛模型车的视频、图像资料，并将相关内容编纂收录在组委会出版论文集中。

参赛队员签名：邹放达 魏泽洋 林锐东

带队教师签名：赵锋

日期：2022 年 8 月 19 日

目录

引言

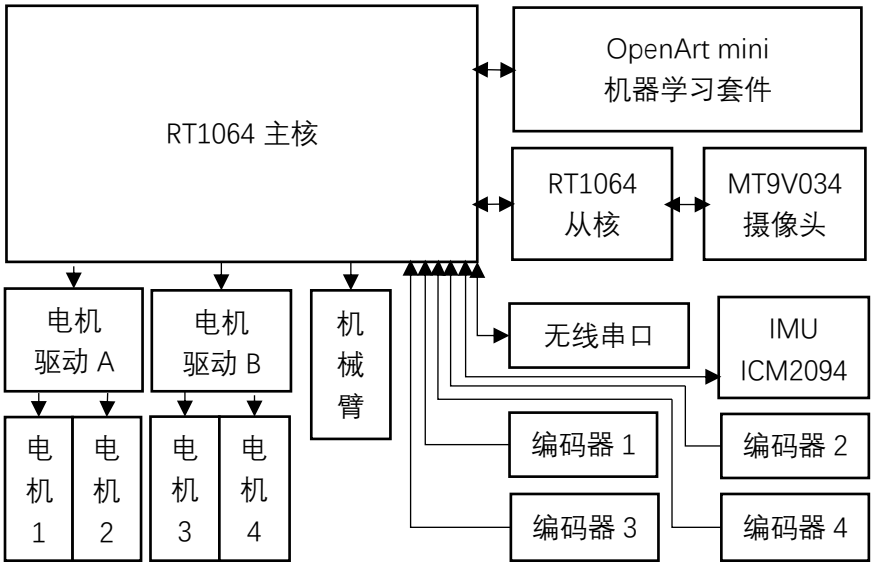
随着现代科技的飞速发展，人们对智能化的要求日益渐增，智能车辆是一个集环境感知、规划决策、多等级辅助驾驶等功能于一体的综合系统，它集中运用了计算机、现代传感、信息融合、通讯、人工智能及自动控制等技术，是典型的高新技术综合体。目前对智能车辆的研究主要致力于提高汽车的安全性、舒适性，以及提供优良的人车交互界面。近年来，智能车辆已经成为世界车辆工程领域研究的热点和汽车工业增长的新动力，很多发达国家都将其纳入到各自重点发展的智能交通系统当中。

在全国大学生智能汽车竞赛中，以智能汽车为研究背景的科技创意性制作，是一种具有探索性的工程实践活动。该项比赛已列入教育部主办的五大竞赛之一。此项赛事涉及的专业主要有：控制、电子、图像处理及计算机等，能极大地培养学生的动手能力及创新型思维，受到全国各大高校的重视。

在这份技术报告中，本团队通过对整体方案、硬件电路、软件算法、机械结构、调试参数等方面进行介绍，详尽地阐述了我们的思想和创意，智能车的制作过程包含着我们团队以及团队指导老师的团结协作、坚持不懈和辛勤付出，今年也是遇到疫情的第三年竞赛，本团队克服重重困难，挑战自我，从零起步，一个一个脚印走过来，培养了敢于实践的动手能力和大胆创新的开拓精神，为今后的学习和工作打下坚实的基础。在此，要感谢清华大学，感谢他们将这项意义非凡的科技竞赛引入国内；感谢中国矿业大学（北京）机电与信息工程学院对此次比赛的关注，我们的成果离不开学校的大力支持及指导老师的悉心教导，还要感谢的是我们三个并肩作战的组员，不离不弃，从校赛走到省赛再走到国赛，协助、互促、共勉帮助我们走到今天，再接再厉！

第一章　系统整体设计

1.1　系统概述



车模的系统组成框图如图 1 所示。

我们使用了双核主控的方案，在主核执行传感器信息融合、车模位姿推算、电机控制等任务，在从核单独执行运算量较大的图像处理任务。

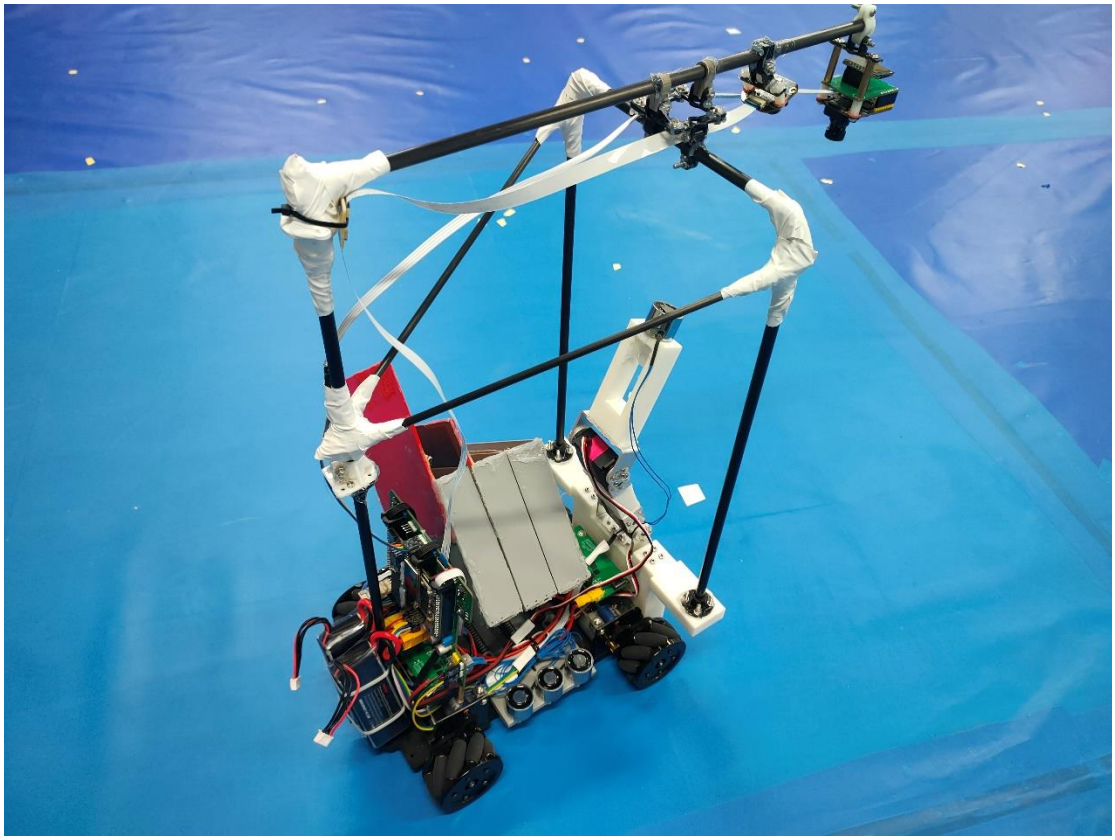
用 MT9V034 摄像头可以获取比赛开始时 A4 纸上任务点的坐标信息，并在车模运行时识别目标卡片相对于车身的位置，对车的坐标位置信息进行修正。使用 ICM20948 模块获取车模运行时 yaw 轴的角速度，并与电机磁编码器所得四个车轮的转速进行融合，可以解算出车模在场地上的位姿信息。基于当前时刻的车模位姿信息和目标点的位置，可以求出对 4 个电机所应发布的速度指令，并使用 pwm 经由电机驱动对电机转速进行控制。

使用 OpenArt mini 识别到卡片的种类后，控制机械臂将卡片抓取并搬运到指定位置，同时使用无线串口将识别结果发送至裁判系统。

第二章　智能车机械结构设计

智能车中的控制都是在一定的机械结构基础上实现的，因此在设计整个软件架构和算法之前一定要对整个车模的机械结构有一个感性的认识，然后建立相应的数学模型。从而再针对具体的设计方案来调整赛车的机械结构，并在实际的调试过程中不断的改进和提高。本章将主要介绍智能车车模的机械结构。

此次比赛选用的赛车车模采用 M 型车模。赛车机械结构只使用竞赛提供车模的底盘部分及转向和驱动部分。整车由 4 个麦克纳姆轮进行驱动。



整车机械机构从前到后以此由电磁铁、机械臂、前置双电机驱动、弹射装置、后置双电机驱动、主控板、电池以及摄像头支架构成。

电磁铁采用的是逐飞科技直径 2.5cm 的电磁铁。机械臂由 2 个 RDS3218 舵机和连接件组成，连接件以及前置舵机支架采用 SLA 光固化 3D 打印制作。前置部分的任务是拾取地面的卡片并将其精准投放到指定的收集框中。在此部分上有安装摄像头框架的孔位。

前置电机驱动由亚克力板打孔安装在车模侧边，亚克力支架板下，利用车模

自有孔位安装电机以及编码器。

车模中间部分是弹射装置，使用 3 个四面盒和配套的转动轴以及弹簧，实现由电磁铁控制的释放操作。

车模尾部亚克力板搭载有后置电机驱动以及主控板。主控板和电机驱动板交错安装。主控板通过铜柱加高并且使用立方体螺母直立安装。电池固定在尾部的摄像头支架杆上，并于主板相连。

摄像头支架采用十字架形式。三个刚性约束使得摄像头尽可能不受震动的影响。

第三章　电路设计

车模使用 2 块 11.1v 锂电池并联供电，并且直接连接在主板上，为满足使用需要，本车模中存在如下几种电压：

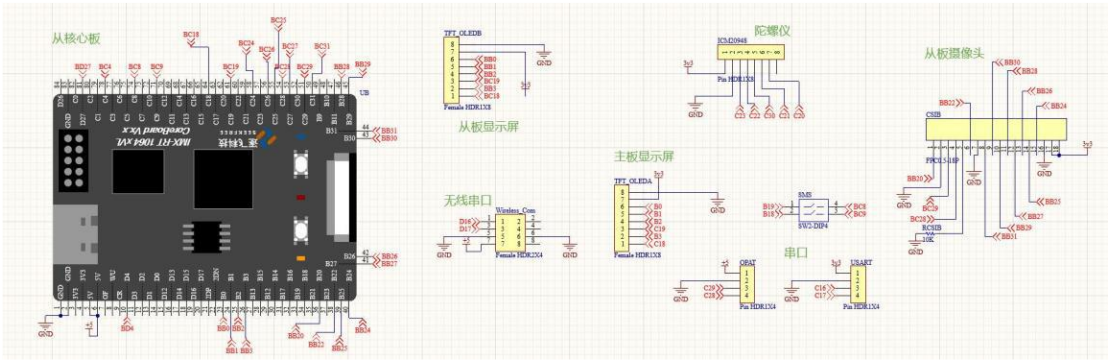
电池电压 11.1v~12v。在电路系统的标识为+12。用来给控制模块和驱动板供电。+12 电压连接到驱动板、开关及稳压电路，当开关按下时由稳压二极管驱动 MOS 管导通 GND 与电池负极，实现电路通电。此处 MOS 管采用 2 块 LR7843。

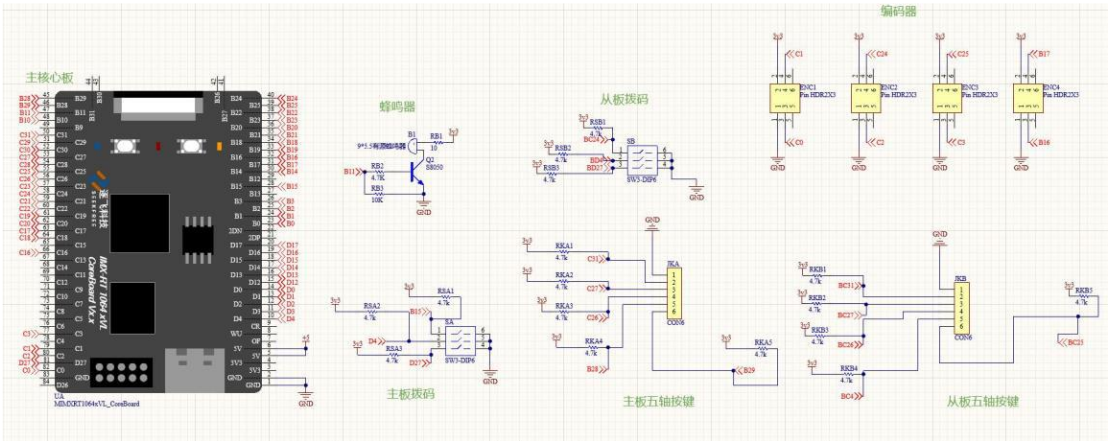
主板阶段电压 6v，通过 TPS54531 将 12v 降压到 6v，给电磁铁以及其他控制电路供电。共有 3 路 6v 电压，另外两路分别给两个舵机供电。

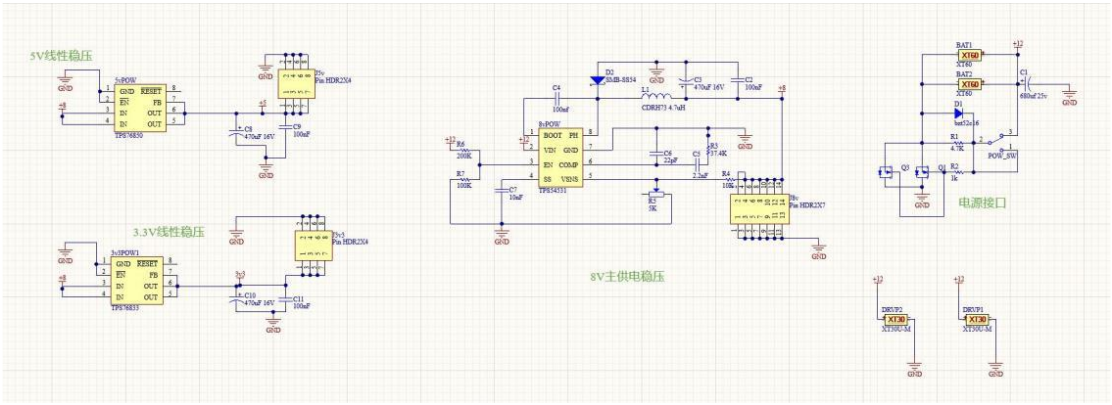
直流 5v 电压，从第 1 路 TPS54531 产生的 6v，经过 TPS76850 低压差稳压器降压到 5v，给单片机等供电。

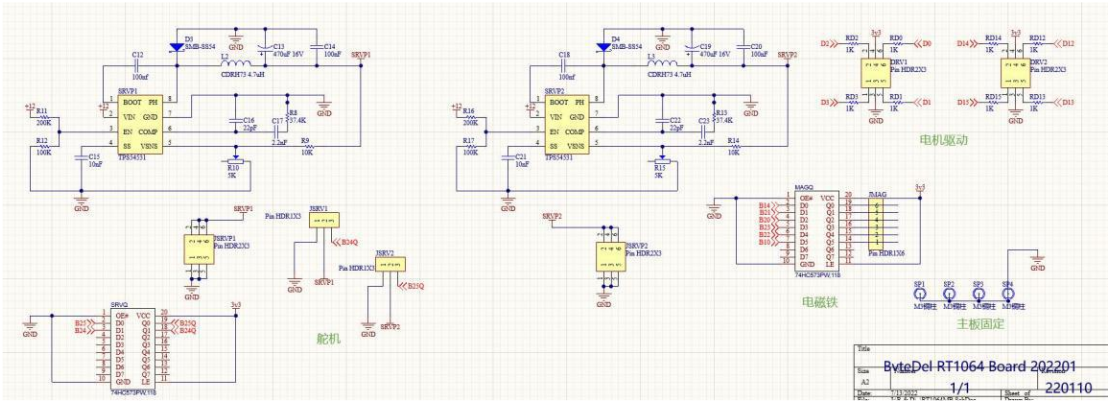
直流 3.3v，从第 1 路 tps54531 产生的 6v，经过 TPS76833 低压差稳压器降压到 3.3v，给摄像头等供电。

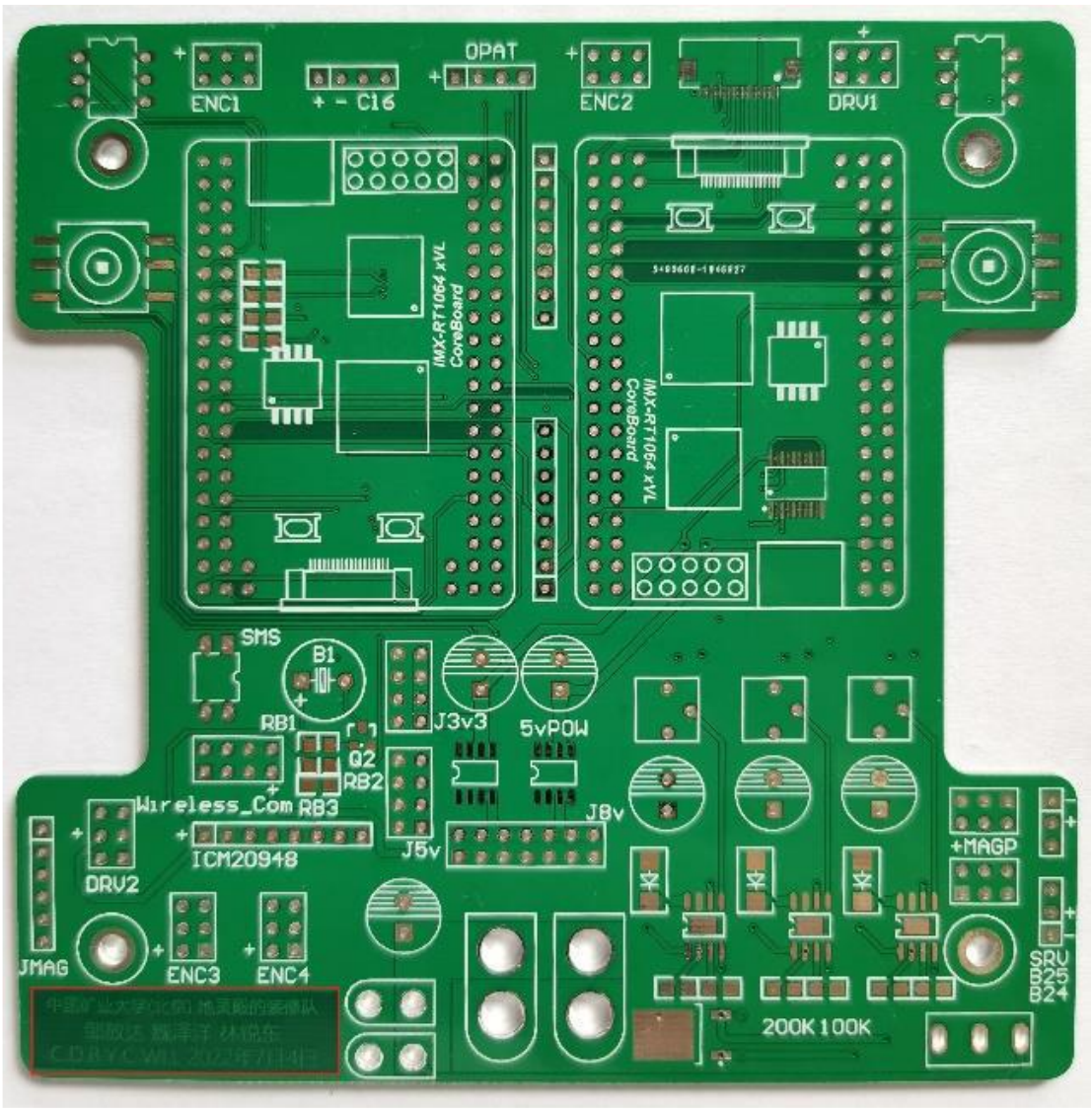
信号部分，分配合适的端口之后采用端到端连接，详细的原理图和实物图如图所示：

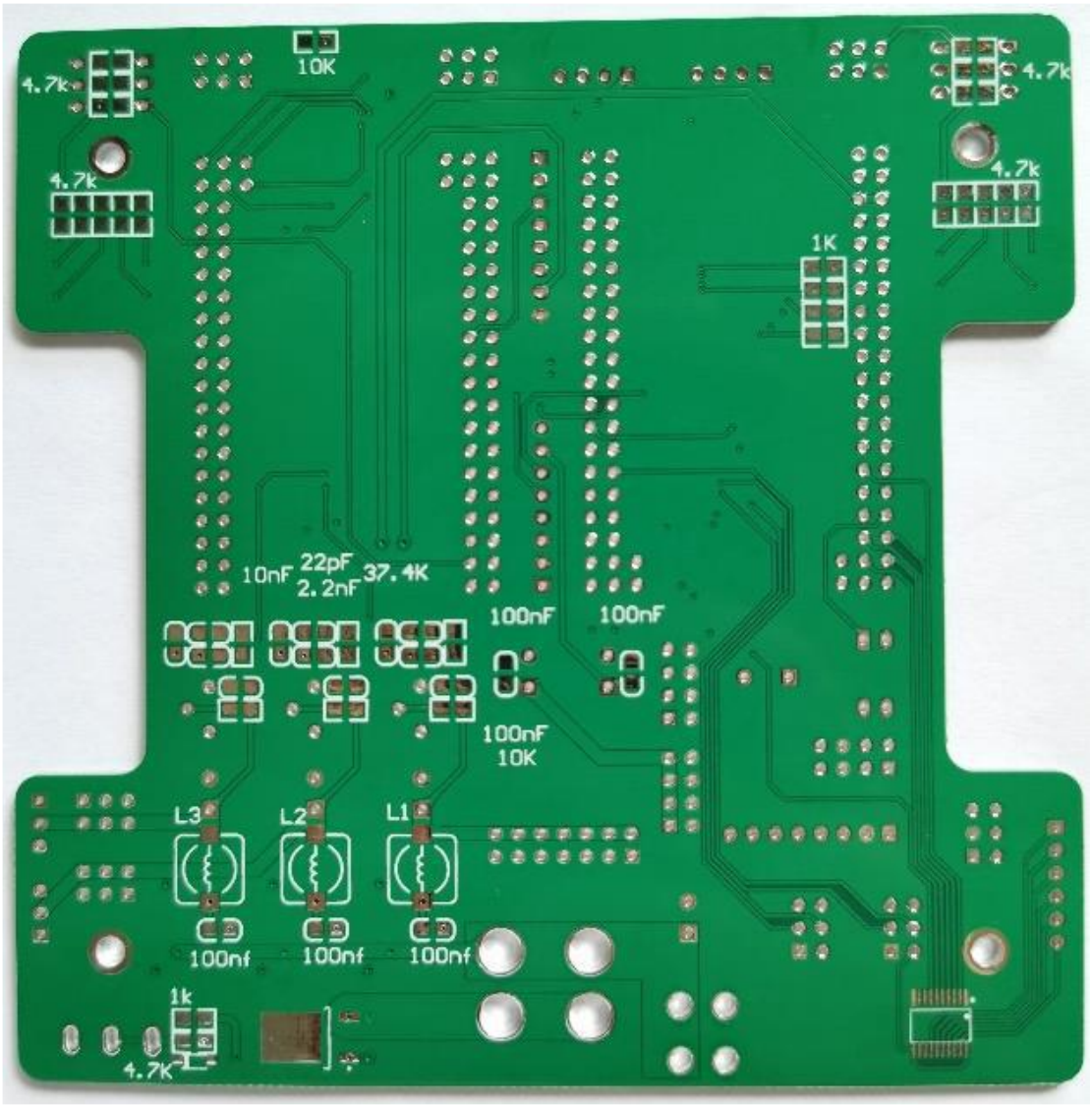




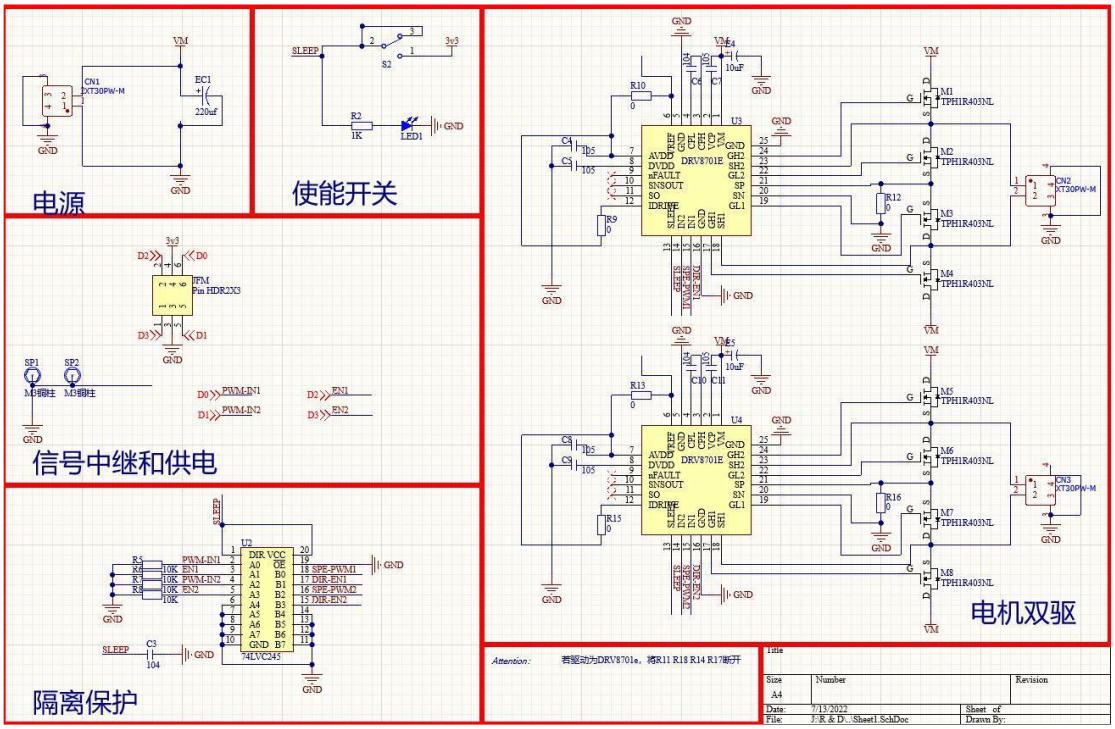

5V线性稳压
3.3V线性稳压
8V主供电稳压
电源接口

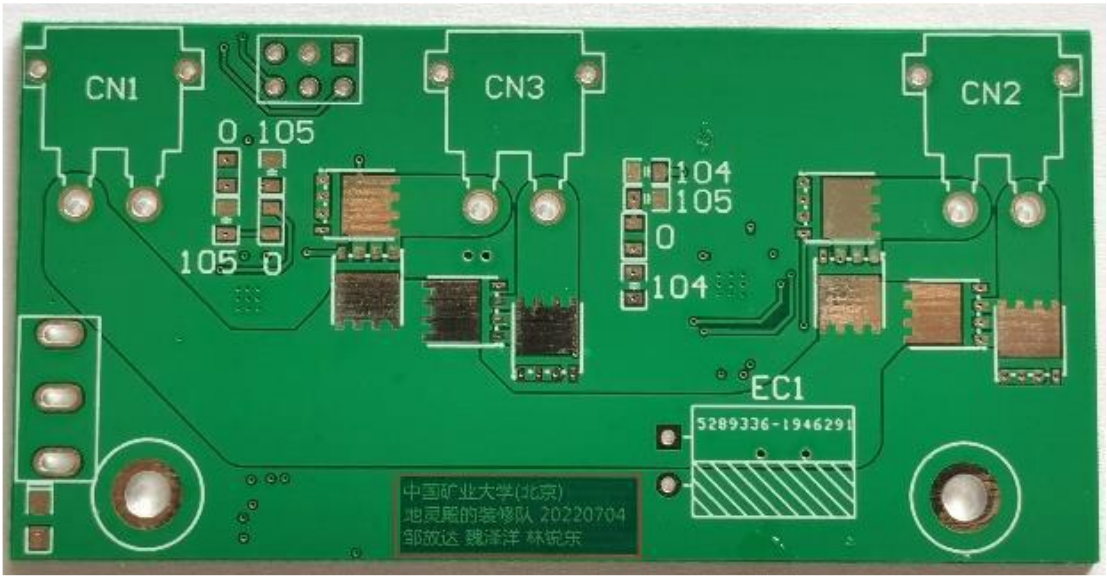

电机驱动
舵机
电磁铁
主板固定
BvteDel RT1064 Board 202201
1/1
220110

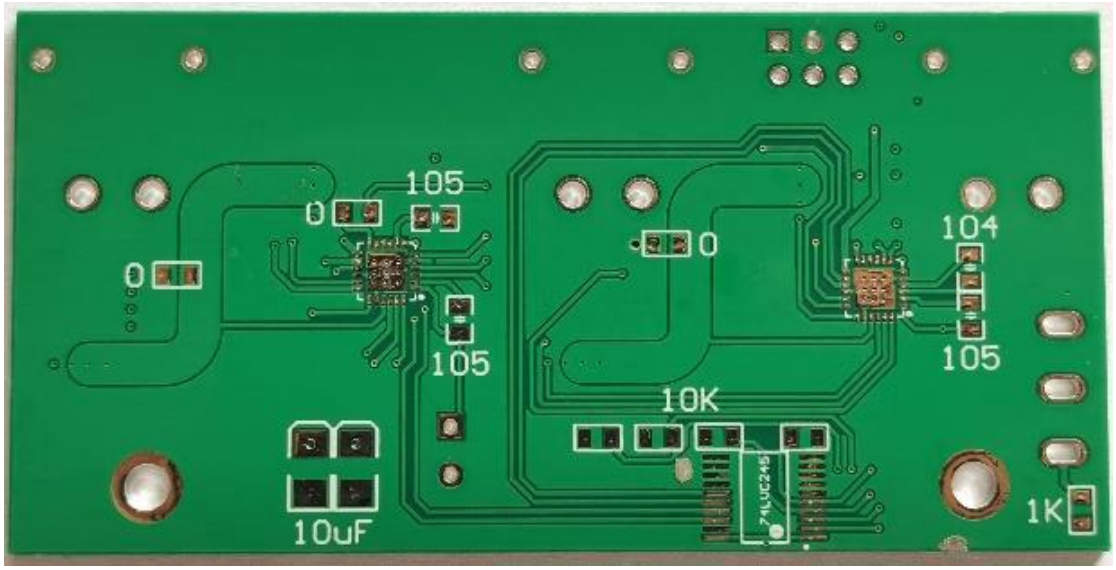

ENC1
+ - C16
OPAT
ENC2
DRV1
IMX-RT1064 xVL CoreBoard
IMX-RT1064 xVL CoreBoard
SMS
B1
RB1
Q2
RB2
RB3
J3v3
5vPOW
J8v
J5v
Wireless_Com
ICM20948
DRV2
JMAG
ENC3
ENC4
+MAGP
SRV
B25
B24
200K 100K



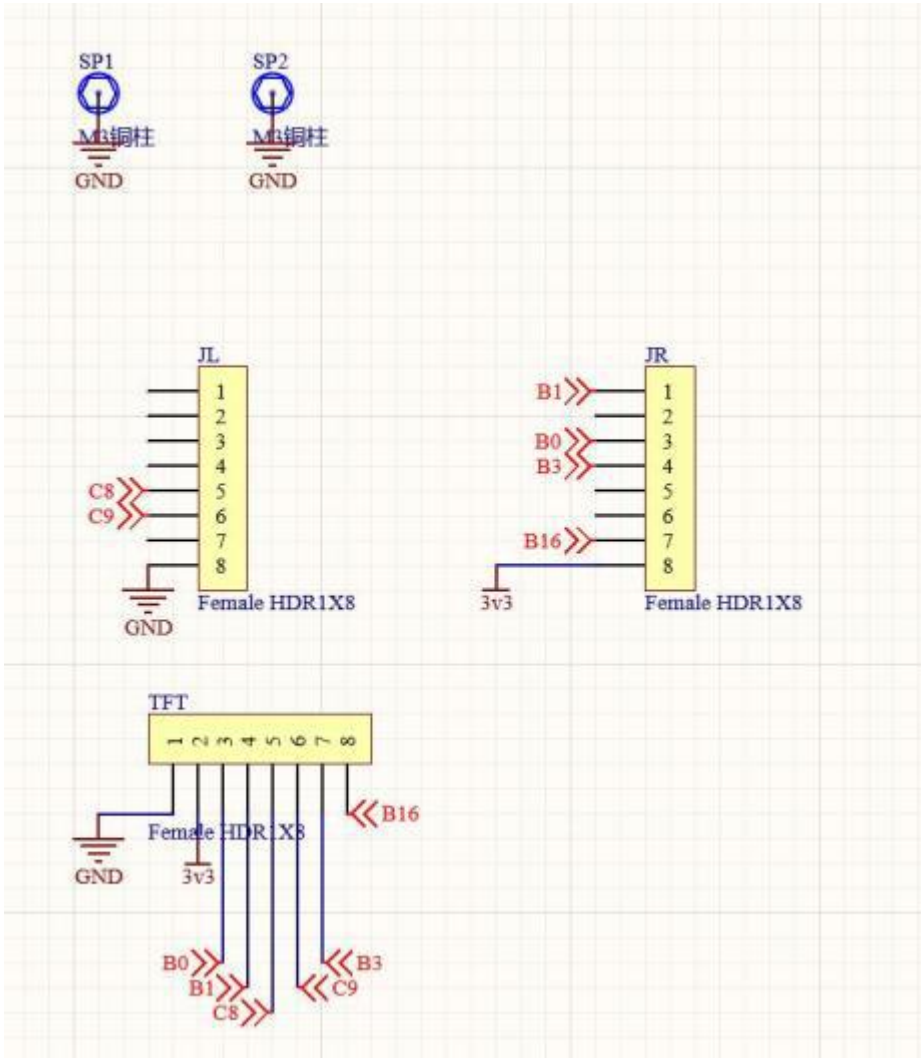
驱动模块，采用 DRV8701E 主控芯片搭配 TPH1R403NL 管组成 4 路电机驱动。详细的原理图如图所示：

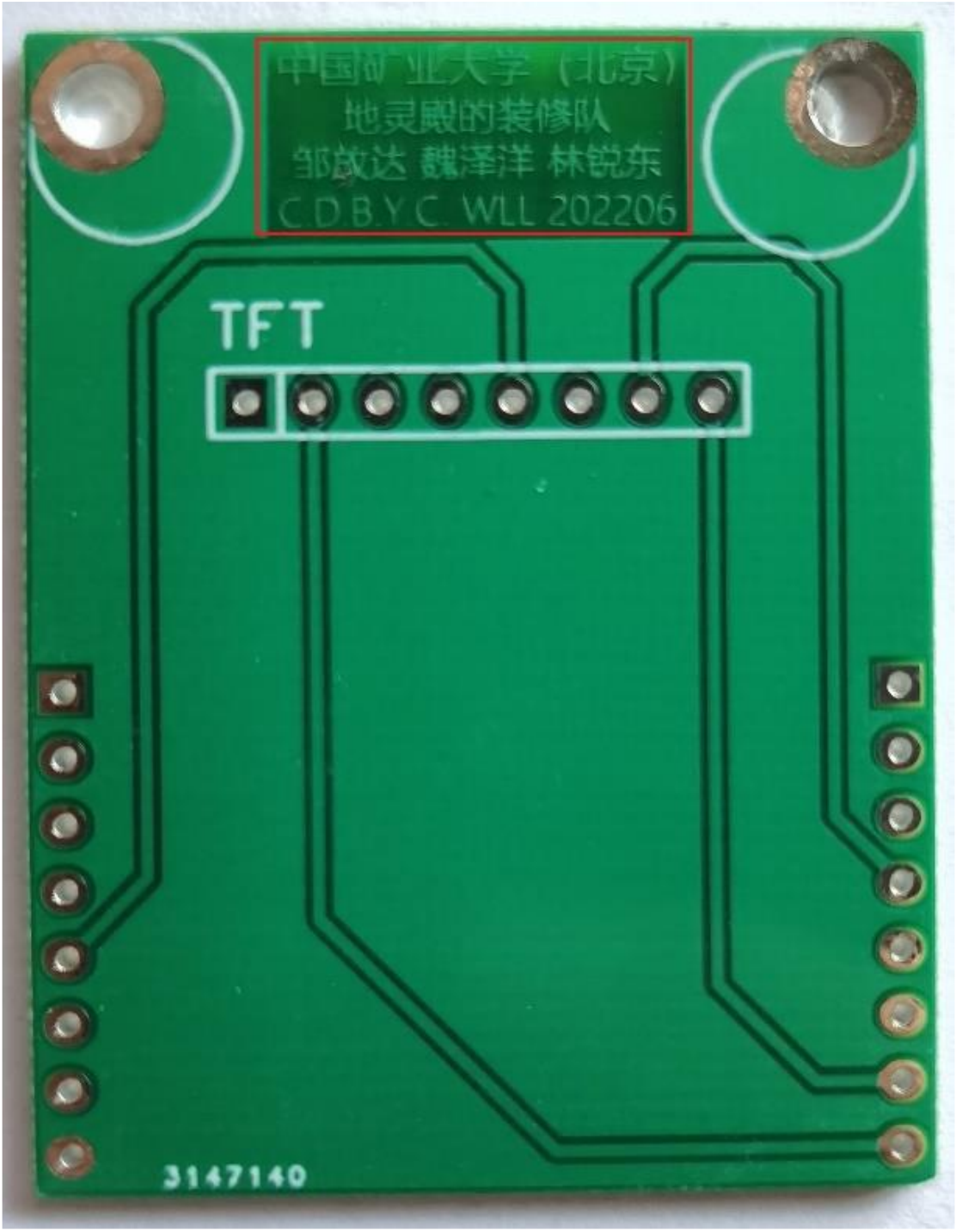

电源
使能开关
信号中继和供电
隔离保护
电机双驱
Attention: 若驱动为DRV8701e，将R11 R18 R14 R17断开

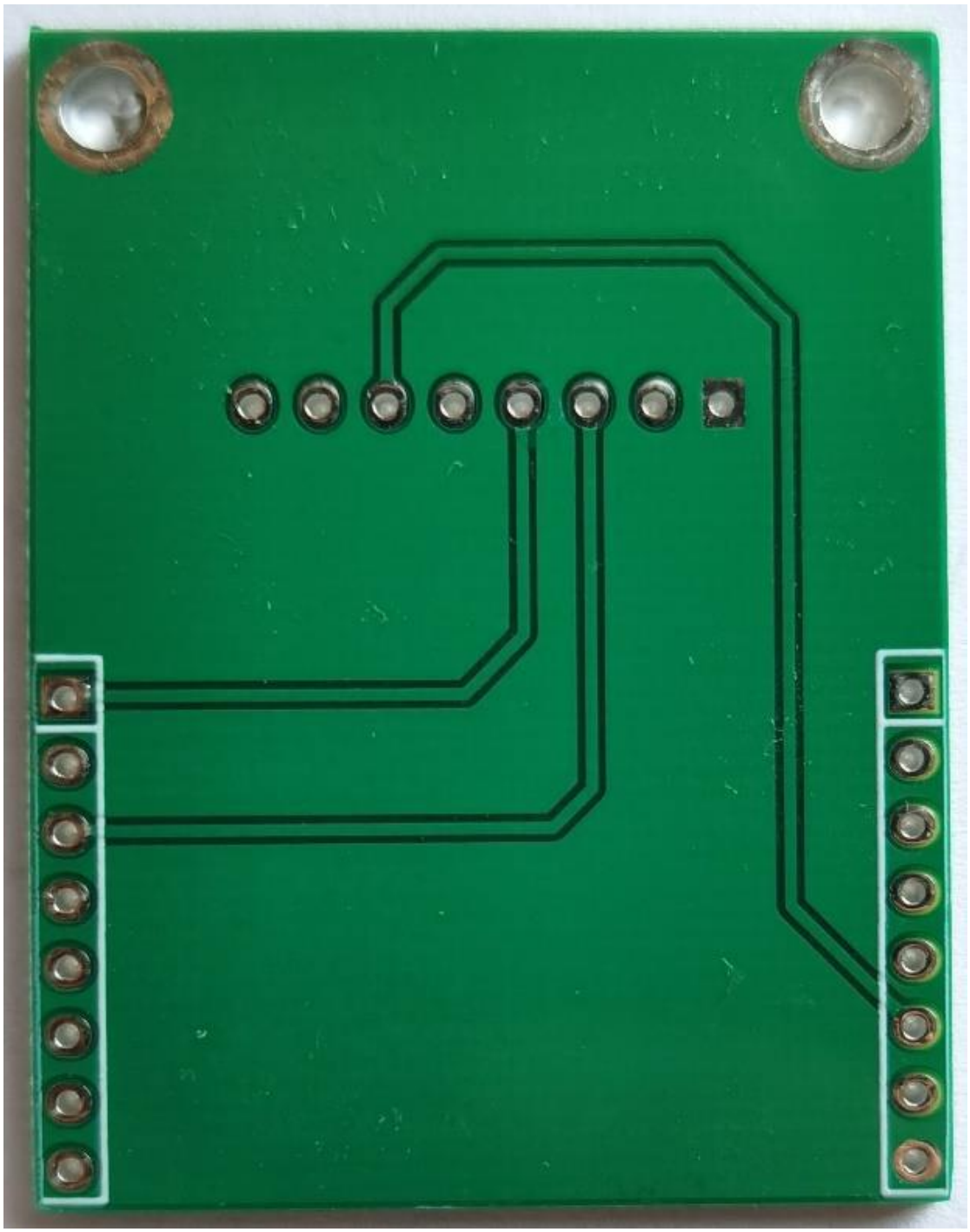

CN1
CN3
CN2
EC1
5289336-1946291
中国矿业大学(北京)
地灵舰的装修队 20220704
邹放达 魏泽洋 林锐乐



摄像头转接板，将 Openart 的部分端口进行布线以安装显示屏




中国矿业大学（北京）
地灵殿的装修队
邹啟达 魏泽洋 林锐东
CDBYC WLL 202206
TFT
3147140



第四章 软件系统设计及实现

我们基于 RT-Thread 实时操作系统，编写了不同优先级的线程用于执行不同的任务。线程按优先级由高到低分别为：

软件定时器中断，用于使用编码器获取 4 个电机的转速，并分别对 4 个电机的转速进行控制。

IMU 数据读取线程，用 SPI 通讯获取车模自转的角速度数据。

串口通讯接收线程，用于与 RT1064 从核、OpenArt 模块以及无线串口上位机的通讯。

main 函数线程，仅以 LED 频闪的方式提示系统运行正常。

Finsh 控制台线程，用于调试。

主运算任务线程，用于传感器数据融合、计算车模控制指令并发布。

任务逻辑线程，用于用户交互，并执行比赛任务。

下面将介绍这些线程的实现细节。

4.1 软件定时器中断

我们使用 M 法测速，在周期为 1ms 的软件定时器中使用磁编码器这次的读数减去上次的读数，除以时间间隔再乘以与车模有关的比例系数，就可以获得车轮的实际转速。

我们使用一阶线性自抗扰控制(LADRC)，通过调整输出到驱动板的 pwm 值来控制车轮的转速，使其能够快速达到并稳定在期望的转速。

同时，我们将四个车轮的实际转速按麦克纳姆轮逆运动学解算求得车模的 xy 方向速度和自转角速度，作为里程计数据存入本地 FIFO 中，供后续使用。

4.2 IMU 数据读取、串口通讯接收

我们结合 RT-Thread 系统提供的信号量和邮箱机制，在 IMU 所产生的 GPIO 中断和串口接收中断中释放信号量或将数据发送到邮箱，在对应的接收线程里获取数据并进行处理，这样即使在对应的线程里需要用到较为耗时的运算或复杂的控制逻辑，也不会占用系统的中断资源导致卡死。

我们选用的 IMU 即 ICM20948 可以设定采样周期，按周期采集数据并产生数

据有效的中断信号。线程在获取到接收信号量后可以使用 SPI 将角速度数据从 IMU 的 FIFO 取回，存入本地 FIFO 中，供后续使用。

我们开了多个串口接收线程，分别用于处理 RT1064 从核所传回的 A4 纸数据和基于目标卡片的位置修正数据，OpenArt 的图像分类数据，以及无线串口上位机的数据。

4.3 main 函数线程、Finsh 控制台线程

main 函数和 Finsh 控制台式 RT-Thread 系统默认自带的线程。

在 main 函数中初始化了所有外设和线程后，直接使用一个死循环来周期性开关核心板上的 LED 灯，用于指示系统在正常运行。

我们没有对 Finsh 控制台进行修改，一般在调试时用它来打印线程运行时的信息。

4.4 主运算任务线程

主运算任务在一个死循环中采用 RT-Thread 提供的线程延时接口来实现周期性执行。

每一轮运算时将里程计数据和 IMU 数据从 FIFO 中取出，使用扩展卡尔曼滤波进行融合，可以得到车模的 3 轴绝对位置和速度，即位姿数据。

通过当前时刻的位姿数据和目标点的信息，以及事先设定参数，如 3 轴最大速度、最大加速度、向前演算时间，可以计算出当前时刻车模的期望 3 轴速度。利用麦克纳姆轮正运动学解算就可以求出 4 个轮子的期望转速。

4.5 任务逻辑线程

我们使用优先级最低的线程来执行按键轮循、显示屏等用户交互相关的任务。在触发发车后，线程将阻塞地执行每个比赛任务，如路径规划、发布目标点等。具体的比赛任务逻辑将放在第六章进行说明。

第五章 视觉识别算法介绍

5.1 find_rects 函数所用算法介绍

find_rects 函数是 openmv 库中的一个函数，可以在图片中寻找与背景有较大颜色差异的四边形。所用的算法实现为 apriltag 算法中初步筛选四边形的算法，可以简单描述为：

将图像转化为灰度图，并对灰度图应用局部自适应二值化。

使用并查集求解二值化图中的连通域，并使用哈希表维护相邻连通域之间连续的边界点。

对于每一簇连续的边界上的点，先对它们的坐标绕重心进行极角排序，并去掉极角值相近且坐标重合的点。

尝试使用最小二乘法拟合四边形的边界。每次用连续的一些点拟合一条直线，得到这些点中间位置的拟合误差。找到拟合误差的一些极大值点，用 4 重循环尝试以这些极大值点为四边形的四个角点拟合四条直线，选择其中拟合误差最小的一组四个顶点作为这个四边形的拟合结果，并根据拟合误差的阈值、相邻边夹教大小等指标粗略筛选掉形状完全不符合要求的四边形。(Apriltag 算法到此为止，下面的两步为 Openmv 中 find_rects 函数的实现)

对于有重合部分的四边形，随机选取其中一个删掉，即去重。

对于每一个结果中的四边形，计算它四条边上像素点位置的图像梯度之和，作为评价四边形拟合结果的标准，并删掉评价值小于设定阈值的四边形。

5.2 对 find_rects 函数的修改

我们将 find_rects 的算法移植到了 RT1064 从核上，并针对所需对函数进行了一定的修改。

5.2.1 求解连通域部分的修改

在求解连通域时，Apriltag 原算法所使用的是八邻域，而在 openmv 中出于性能的考量默认使用了四邻域，八邻域作为可选项并没有打开。我们发现在仅使用四邻域时，如果目标卡片是斜着的，经常出现识别不到的情况，我们将八邻域选项打开后效果显著变好了。

5.2.2 结果去重部分的修改

在 Apriltag 原算法中，是先尝试对四边形内二维码数据进行解码拟合，过滤掉大量不合适的四边形后才进行的去重，而 find_rects 的过滤步骤放在了去重步骤的后面，导致经常有由噪声产生的错误四边形没有被过滤掉，并由于去重的步骤是随机选取删除，导致本来正确的四边形结果被删掉，反而是错误的四边形被保留了下来。

我们去掉了函数的去重和计算梯度和的部分，保留所有的结果四边形，之后将所有得到的四边形逆透视变换到车身坐标系下，依据目标卡片的实际边长和 4 个直角筛选出唯一的正确结果。

5.2.3 对四边形的逆透视变换

在 OpenArt mini 上识别到四边形后，我们结合双三次插值法将所得到的四边形内的部分逆透视变换到原图的左上角，可以得到形式与所公布数据集完全相同的图片形式，因此可以直接对原数据集进行训练，而不需要进行实地拍摄或全角度随机旋转，只需要旋转 90 度的倍数即可，这样做识别的效果也十分精确。

RT1064 从核的任务是利用识别到的目标卡片的四边形推算出目标卡片相对于车身坐标系的位置，因此最终只需要获取卡片像素中心逆透视变换后的坐标即可。将这个结果发送到 RT1064 主核后，主核可以结合当前时刻车身位姿中的偏航角推算车身的准确坐标，从而对车身在地图坐标系下的坐标进行修正。

5.3 A4 纸识别算法的介绍

我们的 A4 纸识别算法只需要满足 A4 纸没有弯曲、A4 只旋转角度不超过±90°，且 A4 纸的边界框可以围住摄像头图像的中心即可，对摆放位置及 A4 纸法线方向与镜头方向夹角几乎没有要求。算法的步骤如下：

使用 Canny 边缘检测算法检测图像边缘，得到存储边缘信息的二值图像。

对边缘二值图像进行 3x3 形态学膨胀操作。

从图像中心点开始，向四周每隔 30 度作一条射线，碰到图像中的距离足够远的黑色像素时停止(防止图像中心有一个目标点，导致所有射线都在中心就停了)，选取其中最长的一条射线与黑色像素的交点作为种子点。如果有任意一条射线到达了图像边缘，说明边界框没有围住图像中心，返回识别失败。

以种子点为起点，用深度优先搜索探索种子点所联通的黑色像素，并存入变长数组。如果搜索到的点数小于阈值或搜索时碰到了图像边界则返回识别失败。

对这些点应用 find_rects 函数中拟合四边形的部分，如果拟合误差过大则返回识别失败。

将边界框上的像素点作特殊标记，并以图像中心为起点进行泛洪搜索边界框内的目标点，如果碰到一团聚在一起的数量大于阈值的黑点则认为识别到一个目标点。如果搜索时碰到了图像边界或某一个黑点的长或宽大于阈值则返回识别失败。

求解边界框所拟合的四边形相对于场地真实长宽的逆透视变换。

将所识别到的所有目标点投影到地图坐标系下，并根据地图方格的尺寸对透视变换所得结果进行对齐。

5.4 图像分类神将网络的训练与部署

我们发现 OpenArt mini 的初始固件并没有将其 32MB 的 SDRAM 内存利用完全，于是通过修改固件的方式使其能够运行足够大的神经网络模型。

我们使用了 Tensorflow 的 Keras 框架中现有的 EfficientNetV2B1 模型，分辨率设定为 96x96，并基于使用 ImageNet 数据集预训练的权重进行迁移学习。由于我们实现了对于目标卡片的逆透视变换，因此只对于数据集进行了光照、色彩、模糊、90 度旋转等较为简单的数据增强，并没有进行随机角度旋转或实际拍摄。训练完之后我们使用 tensorflow 的 TFLiteConverter 对模型进行 int8 量化。最终模型的大小为 8MB，在 OpenArt 上处理一张图片的耗时约为 1.5 秒，几乎不会出现识别错误。

第六章 比赛任务逻辑的实现

比赛任务逻辑放在了优先级最低的线程内，流程为：

发车手按下初始化按键，将 A4 纸放在 MT9V034 摄像头前。

RT1064 从核识别完 A4 纸将结果传回 RT1064 主核，发车手按下确认按键，

触发发车。

主核通知从核从识别 A4 纸模式切换到检测目标卡片模式。

将目标点位间的距离存入邻接矩阵，求解搬运卡片的全局近似最短路径。

按所求路径依次前往目标卡片的位置，并基于从核识别到的卡片坐标实时修正车身坐标。

每到一张卡片前停车后，通知 OpenArt 进行拍照，拍照完成后 OpenArt 通知主核使用机械臂将卡片拾取，在拾取的同时前往下一个目标点的位置。

OpenArt 将所拍照片运行神经网络进行图像分类完成后，再将结果传回主核，主核在独立的线程将识别结果发送到裁判系统，并控制机械臂将卡片放入对应的卡槽。

当场地上所有卡片都拾取完成后，分别前往三条边上进行卡片放置，之后返回车库前的位置。

基于 OpenArt 的 find_blobs 函数识别场地边线，并进行入库。

下面将介绍求解搬运卡片的全局近似最短路径的算法和入库的方法。

6.1 使用 Christofides 算法和 3-opt 算法求解搬运卡片的全局近似最短路径

比赛任务中，车模从车库触发，将所有卡片捡起并搬运到三条边线外后再返回车库的过程，可以建模为度量空间上的旅行商问题。唯一需要特殊处理的步骤是将卡片搬到三条边线外并返回车库的步骤。

设场地长宽分别为W,H，我们将车库的位置$(1,0)$分别沿$x=W$和$y=H$两条边线对称到右上角的$(2W,2H+1)$，并将求出场地中所有卡片到这个点的连线。将连线对称回场地内后即是从某一个卡片出发，分别到三条边放置卡片再回到车库的最短路径。

为了保证旅行商问题求解算法所得到的路径是以这个构造出的车库对称点为终点的，我们又额外引入了一个辅助点，将这个点与起点车库和构造车库的距离设置为 0，与所有卡片的距离设置成无穷大，这样所求出的路径必然是以这个点为终点的。

我们选择使用 Christofides 算法与 3-opt 算法结合求解旅行商问题。

Christofides 算法是一种复杂度较小且近似比优秀的旅行商问题近似求解算法，所求路径长度不会超过最优解的 1.5 倍。算法的流程大致可以描述为：

使用 Prim 算法求解输入图的最小生成树。

选取最小生成树中的奇点，为奇点的子图构造最小权匹配。

将最小权匹配的结果与最小生成树合并，使用 Hierholzer 算法求出合并后的多重图中的一条欧拉回路。

去除欧拉回路中重复遍历的节点，得到的就是算法的输出。

其中计算一般图最小权匹配的步骤一般使用 Blossom 算法进行求解，但由于其复杂度较大，实现较为复杂，且我们后续使用了 3-opt 算法进行了进一步优化，因此我们改用了贪心算法仅求解了近似最小权匹配。

3-opt 算法是一种简单的局部搜索算法，来一条初始的回路中使用三重循环遍历回路的边，并尝试通过重新连接三条边的六个端点的方式来让回路变得更短。我们以 Christofides 算法的输出作为初始回路，使用 3-opt 算法进行优化，取得了不错的效果。并且由于算法的复杂度较低，在具有 23 个路径点的情况下，计算的耗时不到 1 毫秒。

6.2 使用 OpenArt 的 find_blobs 函数辅助入库

由于在拾取最后一张目标卡片后，车模需要先后前往三条不同的边界进行卡片放置并返回车库，期间没有办法可以修正车模的 xy 坐标，车模入库前的坐标经常会有较大的偏移。我们的方案是：

在放置完离车库最近的动物类卡片后，让装在车头的 OpenArt 朝向 x 轴负方向，车模后退进入场地。

当 OpenArt 检测到黄色边线后，车模的 x 轴位置正好位于车库的正中间。这时让车头朝向 y 轴正方向，并后退进入车库。

再次检测到黄线时说明车模已经停入车库，发布停车指令即可。

使用 OpenArt 识别黄色边线的方法是：在图像中心 80x80 的范围内应用 find_blobs 函数，如果返回的黄色像素数高于阈值则判断为识别到黄色边线。

第七章 系统开发及调试工具

代码的编写均在 Visual Studio Code 中完成。VSCode 是一个强大的开源轻量化源代码编辑器，支持语法高亮、代码自动补全、定义跳转等功能，内置了命令行工具和 Git 版本控制系统，又有着丰富的扩展插件生态系统。

对单片机程序的编译、烧录和联调在 Keil MDK 中完成。Keil MDK 为 ARM 处

理器设备提供了一个完整的开发环境，易学易用、功能强大。特别是可以对程序进行单步运行、多步运行、插入断点、观察变量等操作。

对 OpenArt 程序的调试在 Openmv IDE 中完成。它可以将采集到帧缓冲区中的图像显示到屏幕上，并在无需重新上电的情况下实时修改程序源代码并执行。

为了更方便地利用无线串口模块在线观察车模的运行状况，我们基于 python 的 tkinter 可视化编程框架，结合 numpy、matplotlib 等开源库开发了一个串口上位机，具有自定义参数上传逻辑、虚拟示波器、图传、IMU 姿态可视化、车模运行位姿可视化、A4 纸绘制、日志记录等功能。

总结与展望

大学生智能汽车比赛在大学生各类学科竞赛中算的上是战线拉的非常长的了，经过大半年的准备，在调车中也发现了不少问题和困难，然后再一步一步解决改进，尝试过不同方案，比赛中看见其他优秀的方案，看见别人的优点和自己

的不足，进行改正调整解决各种问题。

一个队伍中的三个人各自分工不同，但不代表我们不需要关注了解其他人的工作，软件和硬件是一个相互依存的关系，硬件是软件赖以工作的物质基础，软件的正常工作是硬件发挥作用的唯一途径，所以程序员了解硬件则能更好的发现程序上问题和对硬件的需求有哪些，帮助硬件负责人更好的工作。

我们将两年的青春都放在智能车上面，智能车不仅仅是我们的回忆，更承载着我们的梦想驶向远方。不管最后比赛成绩如何，我们做智能车的初心不会改变，我们对智能车的热爱不会改变，希望智能车越办越好，希望我们变得更强！