第十一届全国大学生光电设计竞赛
东北区赛

方 案 设 计 书

竞赛题目： “迷宫寻宝”光电智能小车（实物类）

竞赛队名： IKUN 队

东北地区大学生光电设计竞赛委员会制

二〇二三年七月

IKUN 队方案计划书

一．总体方案

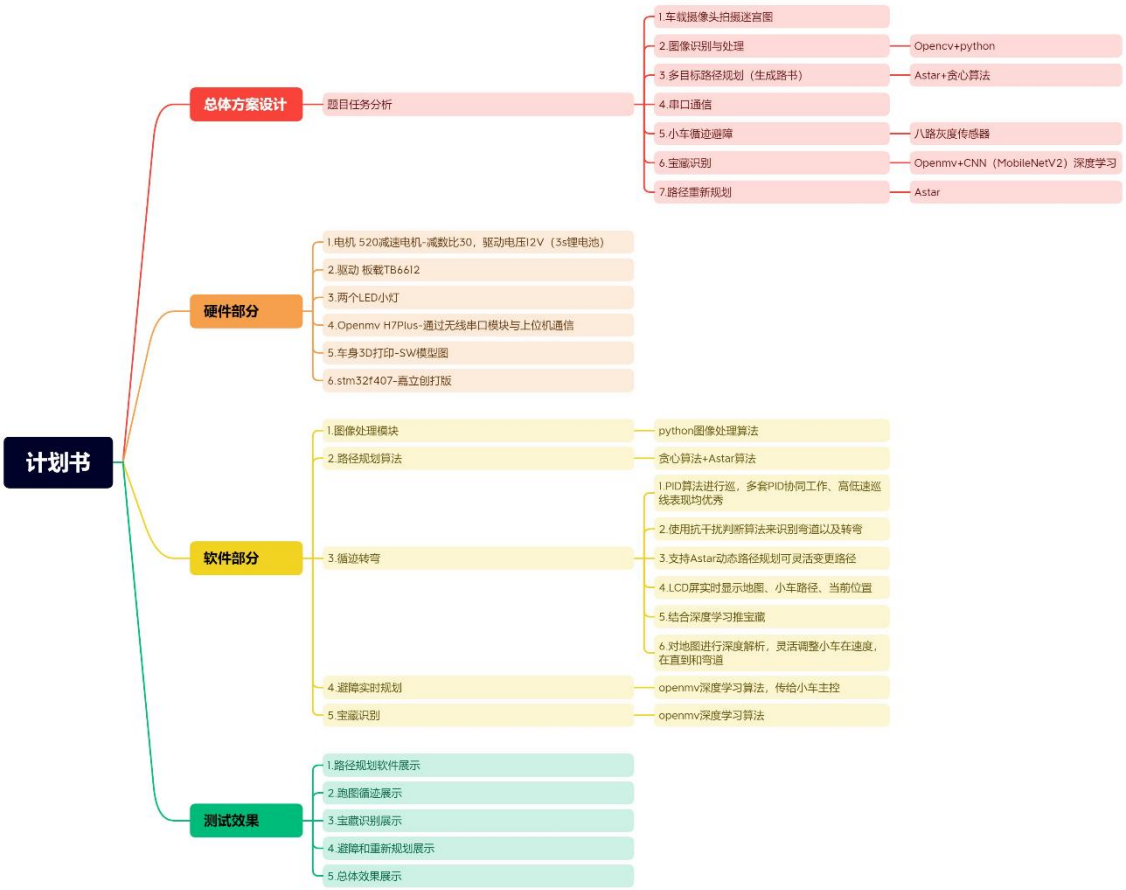

图 1 方案计划导图

本次竞赛旨在综合运用多项技术，包括图像识别、路径规划、循迹避障和光电传感等，以设计制作一款光电智能小车，实现自动寻找并判断迷宫中真伪宝藏，最终成功走出迷宫。

总体方案涵盖了硬件和软件两个方面。在硬件视觉部分，采用 OpenMV 作为图像识别的核心，搭配八路灰度传感器用于循迹，并通过串口与小车 STM32 进行通信。

在软件部分，使用 Opencv-Python 实现高效的图像处理算法。路径规划阶段使用 Astar 算法计算两点间的最优路径，而宝藏探寻顺序则采用贪心算法以提高寻找宝藏的效率。

循迹部分结合八路灰度传感器和 PID 控制算法，实现精确的循迹功能，确保小车能够在复杂的迷宫中稳定行驶。宝藏识别和避障部分利用 MobileNetV2 深度学习网络模型进行判断，提高宝藏判别和避障的准确率。

综上所述，本方案充分发挥各项技术优势，打造一款功能强大且高效的光电智能小车，实现自动寻找并判断迷宫中真伪宝藏，最终成功走出迷宫。各个部分紧密配合，共同实现了小车的智能化探索能力。

二．硬件设计

为了充分贴合本次比赛赛题的要求，我们为小车量身打造了一款适合赛道尺寸与比赛任务的车体，全车车体外观均为本队成员通过 SolidWorks 软件自行绘制，并使用 3D 打印机打印，具有简约的外观，较为灵巧的尺寸，让人眼前一亮。

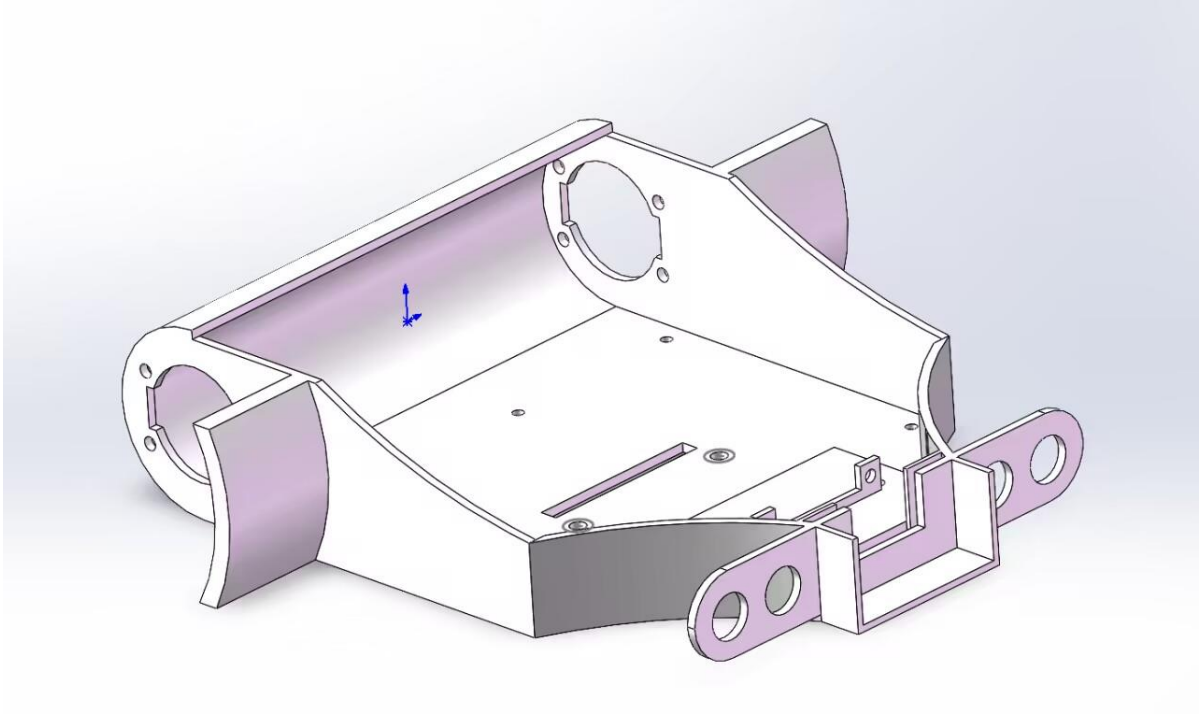

图 2 车体模型图

车身打印采用的是 PETG 材质，PETG 板材具有突出的韧性和高抗冲击强度，其抗冲击强度是改性聚丙烯酸酯类的 3~10 倍，并具有很宽的加工范围，高的机械强度和优异的柔性，比起 PVC 透明度高，光泽好，容易印刷并具有环保优势。第二辆小车的车身材质使用了 ABS 板材，ABS 板材有优良的力学性能，其冲击强度极好，可以在极低的温度下使用；ABS 的耐磨性优良，尺寸稳定性更好。

车身整体设计思路较为清晰，为最大限度利用车身空间，使全车看上去简约、美观，提前测量好每个附件的尺寸大小，使车上的每个附件都能紧凑的安装在车体上，达到了最大空间利用率。

车体前部两边各留有两个小孔，为车灯留下了安装和过线空间，中间留有一个 U 型凹槽，用于装夹摄像头，很好地利用了空间，且向前突出一部长度，既起到保护摄像头，防止摄像头被撞的作用，又能更好地用于推倒宝藏，一举两得。

车体中前部的突起部分是牛眼轮的安装位，且可以安装螺丝螺母，能够根据小车具体的调试过程与结果，更方便的调整牛眼轮的高度位置，从而调整整车的高度。中部还留有一道用于过循迹模块线的窄缝，本车采用感为八路传感器，安装在小车车底。

车体后部留有圆弧形空间，用于安装小车电机驱动，车轮前部安置了一个车轮保护设计，用于防止车轮与对方小车发生碰撞后，车轮受到损伤。

各模块设计效果图介绍如下：

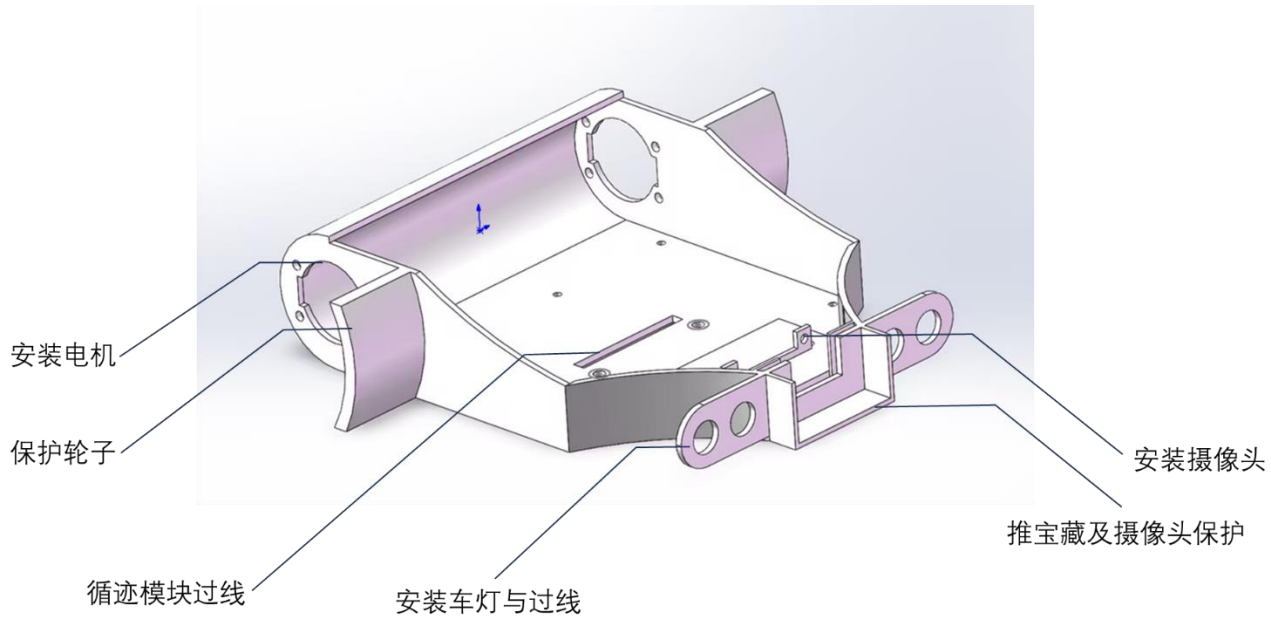

图 3 车体结构

主控芯片采用意法半导体公司的主流芯片-STM32F4 系列芯片，该芯片是一款基于 ARM Cortex-M4 内核的 32 位微控制器，具有较高的性能，较低成本，低功耗，可裁剪等优势。本队充分考虑了各型号芯片的性能及特点，并结合赛题任务要求，选择购置了 STM322F407vet6 的最小核心芯片，并自行在 AD 上绘制了为本队小车定制的拓展版，电机驱动选择了 tb6612 芯片，电路板实物图如下。

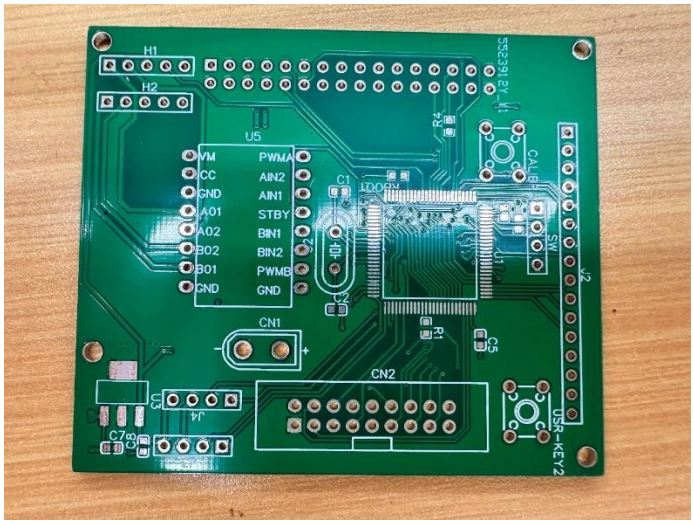

图 4 主控板

所绘制的 pcb 板如下，其中红色为顶层，蓝色为底层：

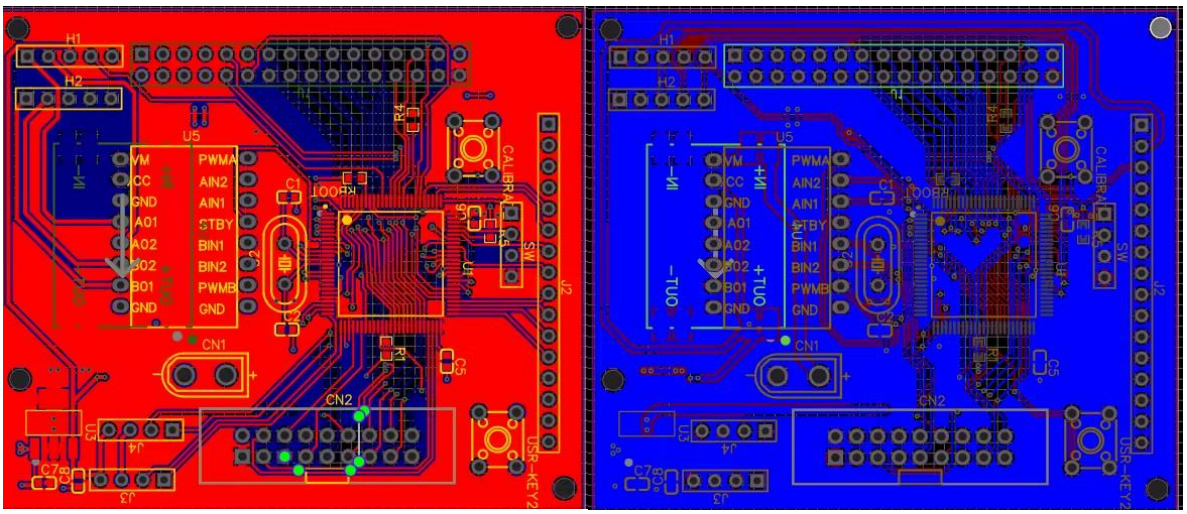

图 5.pcb 板

经过硬件的装配与调整，最终实物效果如下：

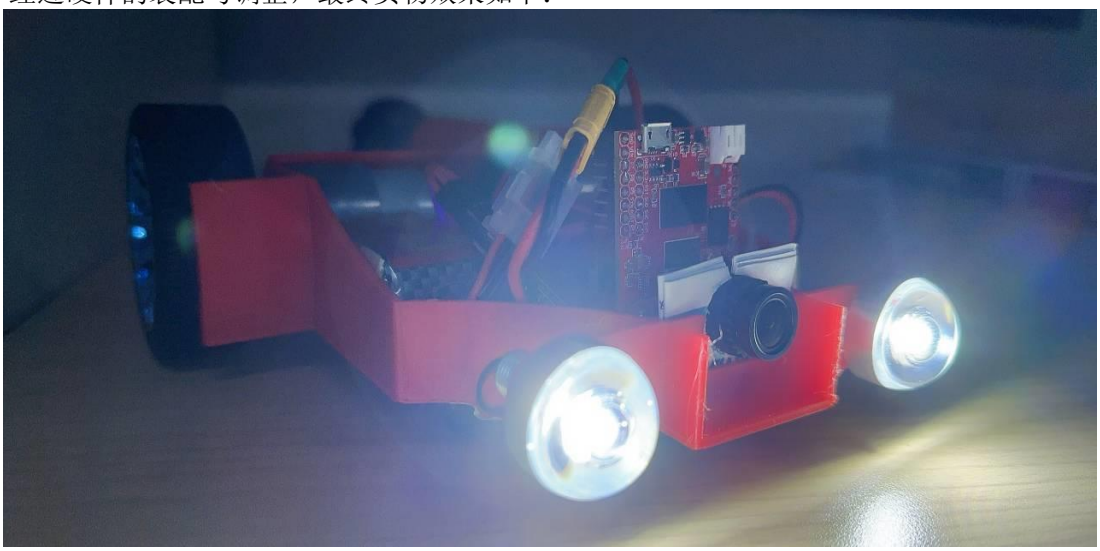

图 6 最终效果图

俯视图如下：

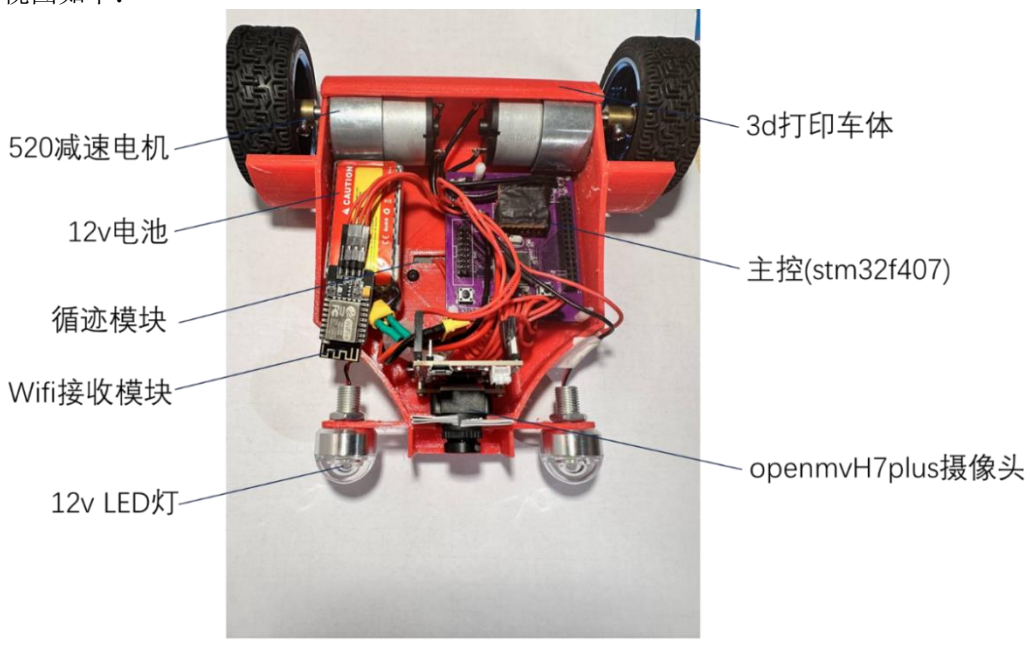

图 7 硬件介绍

三．软件功能实现

1.图像处理模块

1.1 总体流程

项目采用 opencv-python 轮廓识别与轮廓近似法确定方形轮廓，通过分析轮廓关系实现矫正点定位，使用透视变换得到校正后图像。再使用开闭运算以及最小内接正矩形框定地图，最后使用霍夫圆检测确定各个宝藏点的坐标。

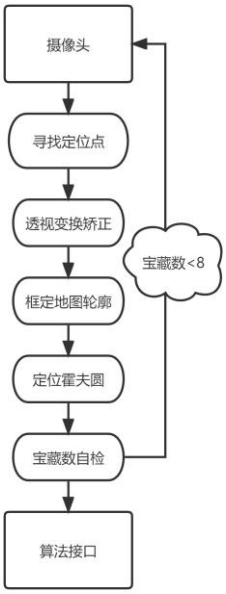

图 8 图像处理算法流程

1.2 寻找校正定位点

首先对原始图像进行灰度处理和高斯滤波，并进行二值化，去除高频噪声点和阴影。其次使用 Canny 边缘检测算法检测图像的边缘，再采用 Opencv 轮廓识别方法 findContours 提取出图中的轮廓，本项目采用轮廓近似法实现各种轮廓的识别，即一条曲线可以由一系列短线段近似，进而可以生成近似曲线，该曲线由原始曲线定义的点子集组成，调用 cv2.approxPolyDP 即可实现。接着对最大轮廓进行多边形逼近，以便获得更精确的轮廓

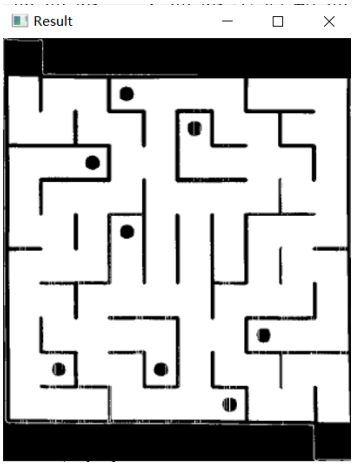

图 9 最大轮廓图

对于提取后的轮廓，使用上述的轮廓检测器提取出满足正方形或长方形条件的轮廓。对于位于四角的定位点，可以发现其内部含有方形小轮廓，故筛选出含有子对象的方形轮廓。又发现其具有并列关系，即都属于某一个较大的父级轮廓，所以通过分析轮廓继承关系树中的父子关系，可以筛选出满足含有子对象且同属于同一对象子对象的轮廓，以上过程使用 findContours 返回值 hierarchy 分析即可。

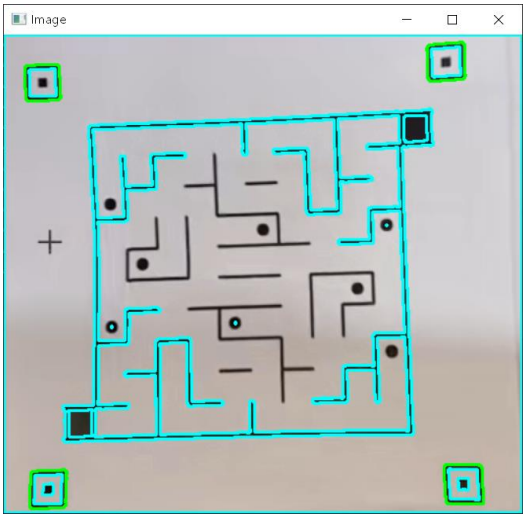

图 10 轮廓搜索

满足以上关系的还有中央的地图轮廓，此时计算五个轮廓面积并去除面积最大的轮廓即可得到四角定位点轮廓，进而使用从 cv2.moments 计算图像的不变矩，通过不变矩的信息可以计算出图形的质心，依次方法的得到是四个质心即为图像进行透视变换的四个定位点。

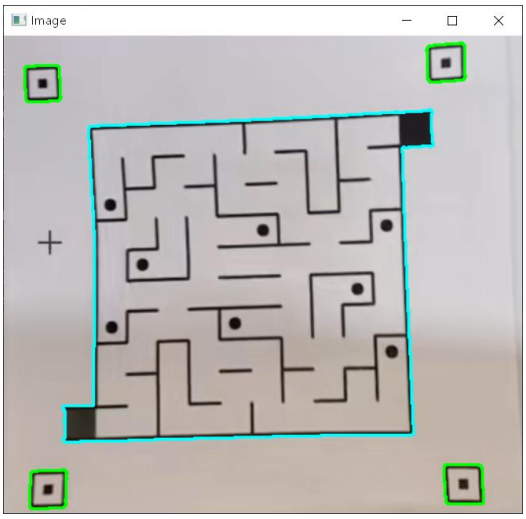

图 11 计算定位点

1.3 透视变换校正

透视校正算法通过求解相机成像的逆变换，将图像恢复到其在三维空间中的原貌。具体而言，需要确定四个点分别代表图像中一个矩形的四个顶点，然后根据这四个点在真实世界中的位置计算出相机成像矩阵的逆矩阵，对图像进行变换即可得到去除透视畸变的结果。

变换公式为：

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{1.3.1}$$

由于分析对象为二维图像，令在 z、Z、a_{33} 值为 1，得到方程：

$$\begin{cases} a_{11}x + a_{12}y + a_{13} - a_{31}xX - a_{32}yX = X \\ a_{21}x + a_{22}y + a_{23} - a_{31}xY - a_{32}yY = Y \end{cases} \tag{1.3.2}$$

求解上述上面含有 8 个未知量（a）的方程,我们需要 8 个像素点，原图像 4 个，即原图像四个顶点；新图像 4 个，即上文分析出的四个标志点。

变换后图像如图所示：

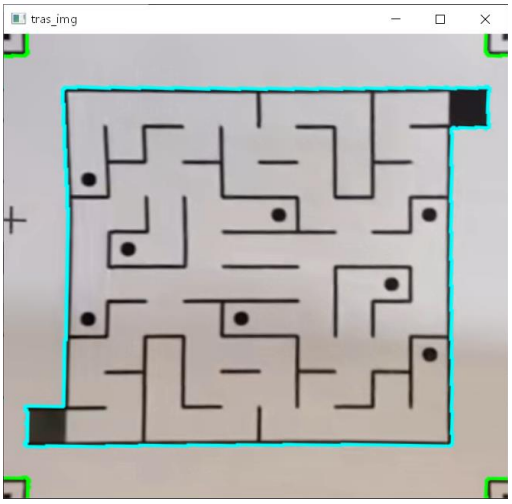

图 12 校正后图像

1.4 框定地图轮廓

得到透视变换校正畸变后的图像后，需要框定地图轮廓并裁剪图像至地图大小方便进行宝藏点坐标相对位置的确定。

本项目采用开闭运算法实现地图轮廓的提取。首先使用 numpy 定义运算核矩阵，由于地图内黑色较为密集，采用先膨胀再腐蚀的操作使地图形成整体，再使用先服饰再膨胀的操作去除不属于地图的分支。最后沿用上文的方案，即先采用灰度处理和高斯滤波，并进行二值化，定义轮廓检测器筛选出唯一的方形轮廓。

图 13 开闭运算效果

由于获取的方形轮廓不一定为标准的方形，故计算图形的最大正内接矩形，并对矩形的四个顶点进行排序，获取轮廓坐标。最后对图形裁剪进行裁剪即可。

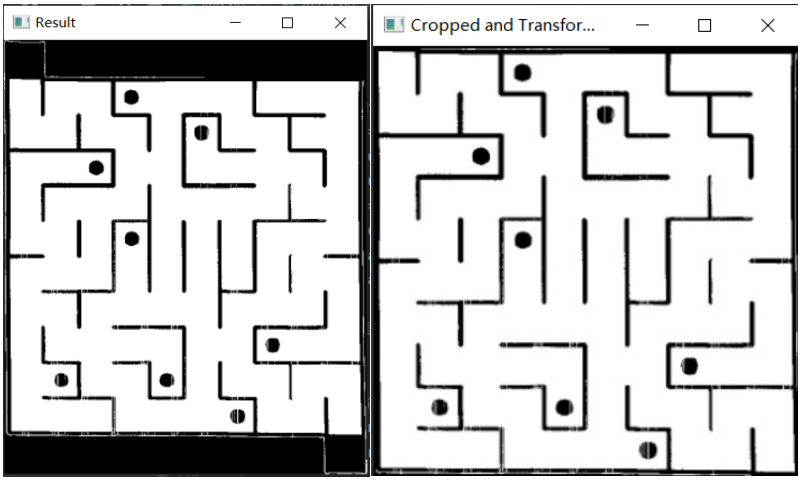

图 14 框定地图效果

1.5 霍夫圆检测

霍夫圆检测法是早期的一种以投票方案进行图形拟合的算法建立霍夫参数三维空间，并对空间内各个单元进行投票，设置阈值从投票结果中筛选合适的圆，并做非极大化抑制，进行调参规定霍夫圆的大小范围后得到 8 个圆心坐标，此时将圆心坐标与方格区间进行对应基于得到各个宝藏点的坐标。

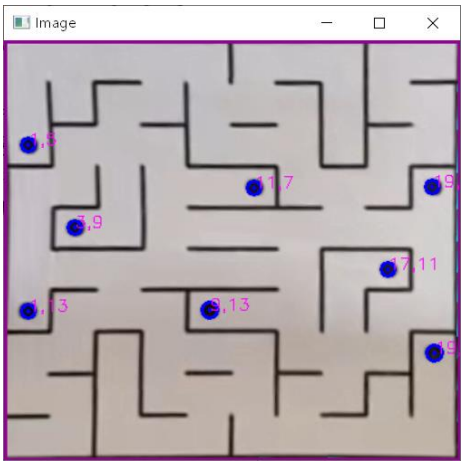

图 15 宝藏点定位效果

2 路径规划算法模块

2.1 地图建模

将迷宫问题模型化，整个地图可以看作中心对称的矩阵，在 2.5 得到的图像基础上对检测到的霍夫圆绘制圆形掩膜，将圆内部区域填充为白色，得到不包含宝藏的地形图，方便接下来的处理。

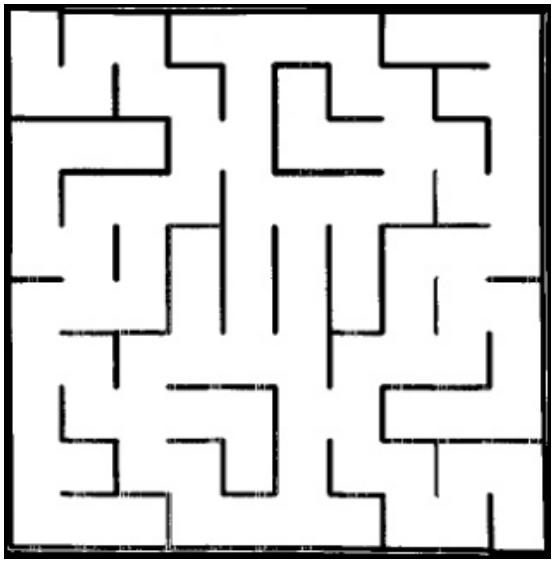

图 16 去宝藏的地图

接下来对迷宫图像进行二值化处理得到“二维码迷宫”，将边界核墙设为 1，可以通行的区域设为 0，依据 opencv 的坐标定位方法，将左上角定为(0,0)，右下角定为(20,20)，起点是(1,1)，终点是(19,19)。

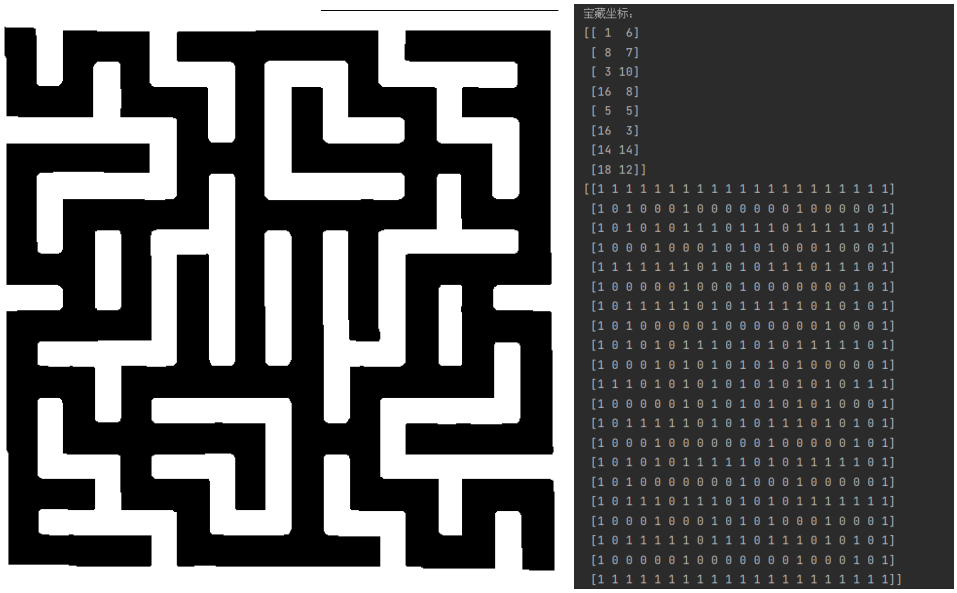

图 17 二值化处理后的地图及地图二维数组

首先需要使从地图一点到达地图另一点时间最短，在理想机械设计条件下，横竖运动速度相等，即路径最短。其次需要满足机器人遍历所有宝藏点以及起止点的路径为所有可能情况下的最优解，即旅行商问题（TSP），TSP 问题无法采用常规数学方法求解，若采用穷举法，耗费时间非常长，故需要设计一种较为智能的搜索算法。

通过以上分析，机器人迷宫寻宝可以看作内层的 Astar 路径规划问题，外层的贪心策略问题。

2.2 内层点对点寻路：Astar 搜索

而 A 算法则是一种启发式搜索算法，通过估计从起始状态到目标状态的代价，选

择下一步要探索的路径。虽然深度优先搜索可以用于解决迷宫问题，但结合 A 算法可以更高效地找到最优路径。以下是使用 A*算法解决迷宫问题的基本思路：。

1）定义迷宫的状态表示。迷宫可以使用一个二维数组或网格表示，其中不可通过的墙壁用障碍物表示，可通过的路径用通道表示。每个状态可以表示为一个坐标点，包括行和列的索引。

2）定义启发函数（Heuristic Function）。启发函数用于估计从当前状态到目标状态的代价或距离。在迷宫问题中，可以使用曼哈顿距离或欧几里得距离作为启发函数，这些距离可以通过当前状态和目标状态的坐标计算得出。；

3）创建一个优先级队列（Priority Queue）用于存储待探索的状态。优先级队列中的元素按照估计的代价排序，代价最小的优先被探索；

4）初始化起始状态，并将其放入优先级队列中。从优先级队列中取出代价最小的状态，并进行以下操作：

① 检查当前状态是否为目标状态，如果是，则找到了解答。

② 否则，扩展当前状态，即根据可行的移动方式（上、下、左、右）生成新的状态，并计算新状态的代价。

③ 将新状态放入优先级队列中。

5）重复步骤 4，直到优先级队列为空或找到目标状态。

6）如果优先级队列为空，说明无法找到解答，迷宫问题无解。如果找到目标状态，可以通过回溯操作从目标状态回溯到起始状态，得到最优路径。

通过以上基本思路，结合 A*算法的启发式搜索，可以实现在迷宫中找到最优路径的算法。

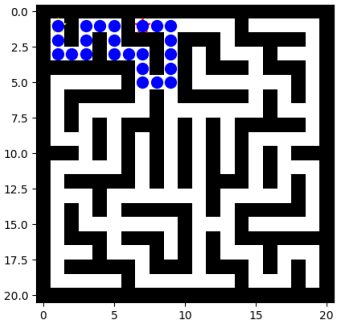

图 18 从起点到第一个宝藏点 Astar 路径规划可视化

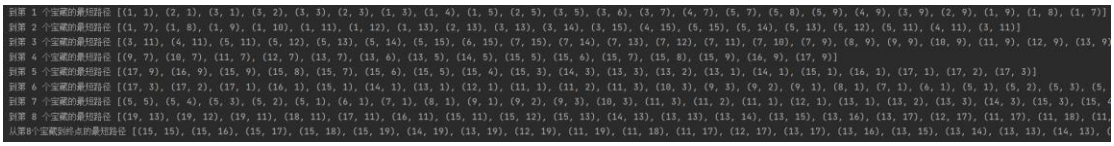

图 19 Astar 算法路径输出

2.3 外层路径规划：贪心策略

对于路径规划问题，贪心算法可以用于逐步选择下一步的移动方向，以期望得到一个近似最优解的路径。下面是使用贪心算法进行路径规划的思路：

1.初始化起始状态，并将其作为当前状态。

2.当前状态为起始状态，选择一个启发规则，根据当前状态选择下一步的移动方向。启发规则可以根据问题的特点进行定义，例如选择距离目标状态最近的邻居状态

作为下一步。

3.根据选择的移动方向，更新当前状态，并将其作为新的当前状态。

4.重复步骤 2 和步骤 3，直到到达目标状态或无法继续移动。

5.如果到达目标状态，则找到了解答，可以根据实际需求进行后续处理。

6.如果无法继续移动，说明无法找到解答，路径规划失败。

贪心算法在每一步中选择当前最优的移动方向，但它没有考虑全局最优解，可能会陷入局部最优解。因此，贪心算法不一定能找到最优解，但它可以在较短时间内找到一个近似最优解。

本方案将当前宝藏点到下一宝藏点的路径距离作为启发规则，路径距离由 Astar 距离进行计算，由此依次确定寻宝顺序，实现外层路径规划。

```
# 计算两个宝藏点之间的路径距离
def path_distance(point1, point2):
    path = astar(maze, point1, point2)
    return len(path) - 1 if path else float('inf')
```

图 20 贪心策略的启发规则定义

```
随机遍历顺序: [(1, 1), (1, 7), (9, 7), (3, 11), (17, 9), (5, 5), (17, 3), (15, 15), (19, 13), (19, 19)]
贪心策略前总的 cost: 240
贪心策略遍历顺序: [(1, 7), (3, 11), (9, 7), (17, 9), (17, 3), (5, 5), (19, 13), (15, 15)]
贪心策略后总的 cost: 180
```

图 21 贪心策略前后总路径 cost 对比

可见贪心策略算法可以极大的避免冗余重复的路径、减小移动路程，提升搜索效率。输出的每一步路经通过串口发送至机器人，写入机器人主控 flash 中。

3.相遇时的策略：避障和重新规划

本小组基于 OpenMV H7plus 摄像头，采用 MobileNETV2 神经网络训练的方法来进行对方小车的检测，从而配合己方小车实现避障和重新规划的功能。

3.1 图像数据采集

由于比赛对小车外形没有要求限制，为了增大前方小车检测的准确性,我们在场地拍摄模拟小车相遇的基础上，从网上搜集了 1000 张各式小车的图片，并利用 Python 程序采用旋转、翻转、模糊、增加噪声、改变亮度等方式对图片进行数据增强得到了 6043 张样本图片，并将 80%作为训练集，20%作为验证集。

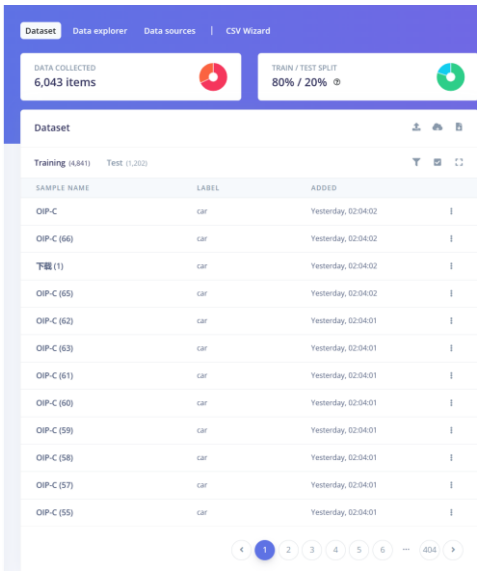

图 22 样本数据

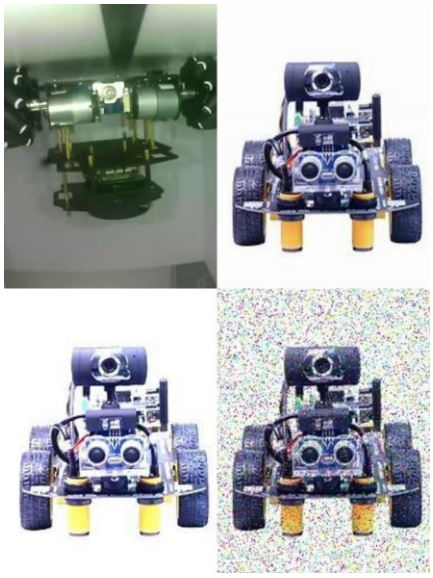

图 23 数据增强示例

3.2 模型建立与训练

本组采用 MobileNETV2 模型进行迁移训练，图片输入大小为 96×96，训练周期数设为 35，学习率设置为 0.0005，Dropout 设置为 0.1。训练结果如下：

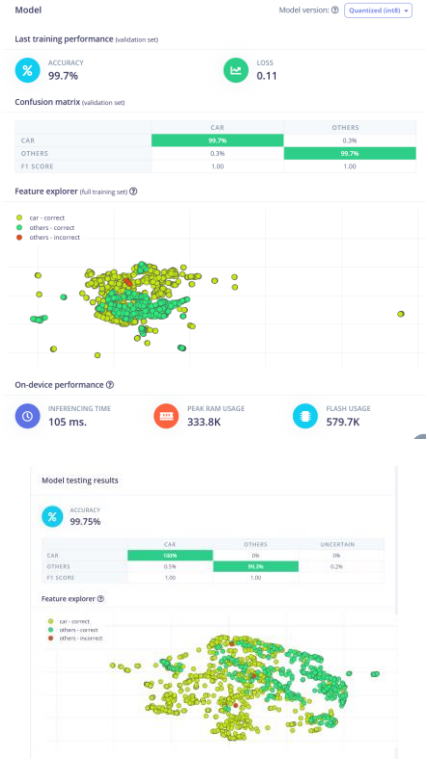

图 24 模型训练结果

由图可知，小车检测的验证集正确率可达 99.75，模型准确率较高，符合比赛精度要求。

3.3 模型部署与实现

本组将训练好的模型导出为 tflite 格式部署到 OpenMV 上，摄像头通过一个 Gpio

口接受小车的输入，小车在运行时摄像头一直拍照分析前方是否存在小车，当小车在拐弯时停止接受摄像头反馈以保障小车转弯程序的稳定性，小车在直道时接受摄像头的检测信息。摄像头再通过另一个 Gpio 口向小车反馈前方道路信息，从而实现小车检测和避障程序。

3.4 策略选择

由于比赛不确定性，本小组对小车相遇时的情况准备了两种策略。

1）避障：如果摄像头及时准确检测到了前方小车，并将信息反馈给了己方小车。己方小车调取地图信息，并将前方路况标记为墙，在车载单片机内重新进行剩余宝藏的路径规划。如果重新路径规划的结果为空，也就标志着己方小车被对方小车堵在了死角，此时小车停止运动进入待机模式，直到摄像头不再检测到对方小车。

2）重新规划：如果小车未能及时检测到对方小车，而被撞出赛道无法继续比赛，本小组会申请从原点重新开始，通过一键重启装置自动规划未探索的路径路径。

4.宝藏识别

本小组基于 OpenMV H7plus 摄像头，采用 MobileNETV2 神经网络训练的方法对宝藏类型进行识别。相比颜色组合和形状识别而言，深度学习方法能够有效辨别在不同亮度环境和拍摄角度下的宝藏类型，识别准确率较高，受场地环境影响较小。

4.1 图像数据采集

本小组模拟比赛场景在实际场地用 OpenMV 摄像头分别收集了红真、红假、蓝真、蓝假四种宝藏类型的图片各 150 张，然后利用 Python 程序采用旋转、翻转、模糊、增加噪声、改变亮度等方式对图片进行数据增强，得到了 6832 张样本图片，并将 80%作为训练集，20%作为验证集。

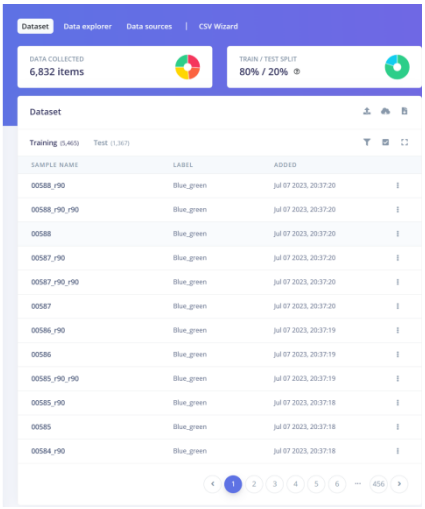

图 25 样本数据

5.2 模型建立与训练

本组采用 MobileNETV2 模型进行迁移训练，图片输入大小为 96×96，训练周期数设为 40，学习率设置为 0.0005，Dropout 设置为 0.1。训练结果如下：

图 26 模型训练结果

由图可知验证集的正确率为 95.9%，在一次单独的模型训练中准确性可达 100%，符合宝藏识别的精度要求。

。

5.3 模型部署与实现

本组将训练好的模型导出为 tflite.格式部署到 Openmv 上，通过一个 GPIO 口接受小车的输入，当小车到达宝藏地点后，小车停止，摄像头进入宝藏识别程序对宝藏进行静态识图判别，再通过另一个 GPIO 口向小车反馈宝藏类型，从而实现宝藏识别。

图 27 模型部署与实现

四．最终效果呈现

表 1

时间/宝藏数	3	4	5	6	7	8
第一次	1min06s	1min20s	1min30s	1min42s	1min52s	2min02s
第二次	1min10s	1min26s	1min36s	1min40s	1min48s	1min59s
第三次	1min15s	1min27s	1min34s	1min48s	1min45s	1min50s
第四次	1min19s	1min24s	1min36s	1min35s	1min42s	1min46s
第五次	1min12s	1min29s	1min40s	1min41s	1min55s	1min53s
平均时间	1min12s	1min25s	1min35s	1min41s	1min48s	1min54s

以上表格是本小组在八种宝藏摆放类型下分别进行五次跑图的时间统计，其中表格第一排代表小车找到所有真宝藏时已探索的总宝藏数。由计算可知 30 次完整跑图

的平均时间越为 1min36s，后续我们将在弯道转弯速度、宝藏识别速度、更合理的宝藏探索顺序算法方面进行优化。

五、结语：

我们团队将全力以赴，充分发挥团队成员的技术和创新能力，通过图像识别、路径规划、自动驾驶、避障循迹、光电传感等技术的综合应用，设计制作一款出色的"迷宫寻宝"光电智能小车。我们相信，在竞赛中我们能够展现出团队的实力和创造力，并取得优异的成绩。

谢谢各位评委的审阅和支持！

2023.7.15

IKUN 队