中国科学技术大学

的卫队技术总结文档

队伍名称：_____的卫队_____

系别（全称+代号）：_____网络空间安全学院，221 院_____

类别：

☑冰壶机器人

□变形机器人

目录

1. 队伍简介

1.1 队名介绍

“的卫”即是“对”的拼音，是小组成员对正确的追求，表现了我们组严谨细致高要求的作风，也表现了我们组希望能够不出错误地完成比赛。同时也巧妙利用谐音，使队名幽默风趣，体现我们组在工作中也不失乐趣，表现了乐观向上的队伍氛围。获奖并非我们组的绝对目标，能走到最后即是胜利，若能获奖便是锦上添花。

1.2 成员介绍与分工

队长	梁郅卓	安排并协调队员的工作，组织组会讨论，参与设计机器人的底盘结构和取壶结构，参与绘制机器人 3D 图像，做好小组规划。
队员	张席正	参与设计机器人的抓取结构和发射结构，参与绘制机器人 3D 图像，负责采购部分零件。
队员	沈新迪	参与进行电路及程序设计，并进行调试。
队员	陈鹤影	整体电路设计连接和分电设计及主控电源等模块的选型工作。
队员	杨城昊	参与进行视觉设计以及算法优化。

2. 机械部分

2.1 功能与结构概述

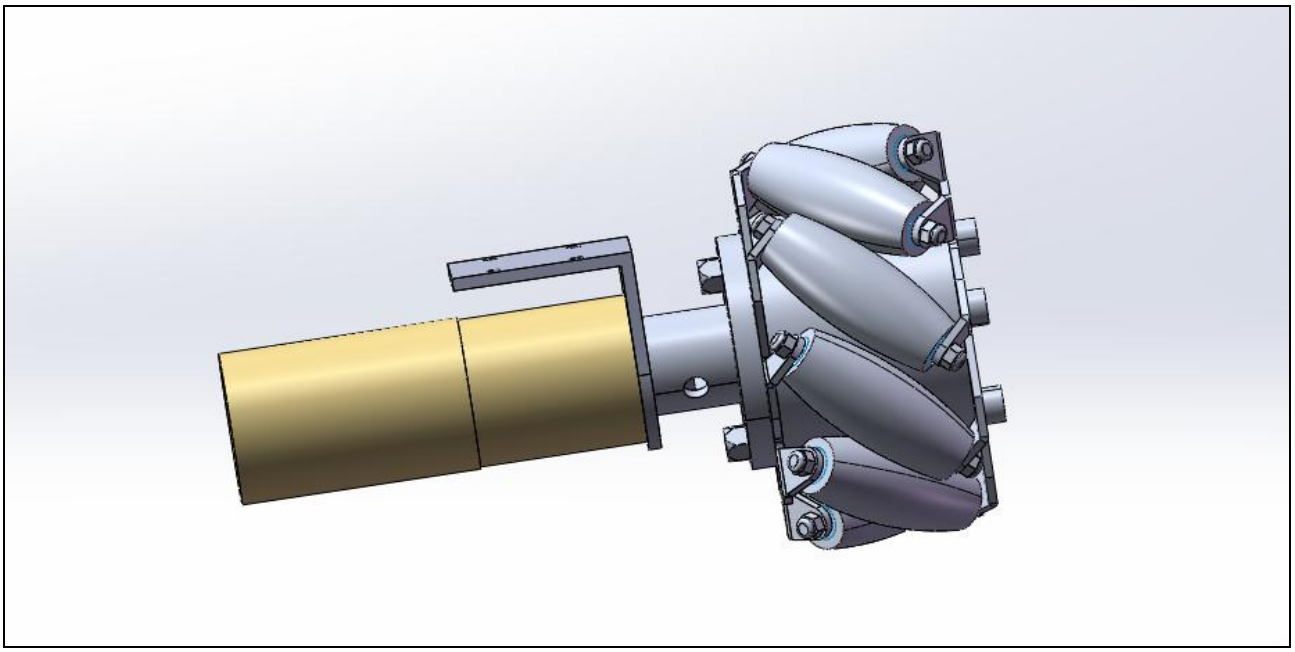
为了完成比赛项目，我们组的机器人需要如下功能：

1. 自动巡线以及精确的测距功能
2. 能精准识别敌我冰壶
3. 有精准的抓取冰壶能力
4. 有稳定的发射冰壶的能力
5. 自动完成整个比赛流程的能力
6. 巡线使用光敏传感器实现，测距功能用激光测距模块实现。
7. 用颜色识别及条形码识别来辨别敌我冰壶。
8. 采用机械爪加电推杆的组合抓取冰壶。
9. 利用气缸推动的方式发射冰壶。
10. 利用单片机及树莓派实现对机器人的自动控制。

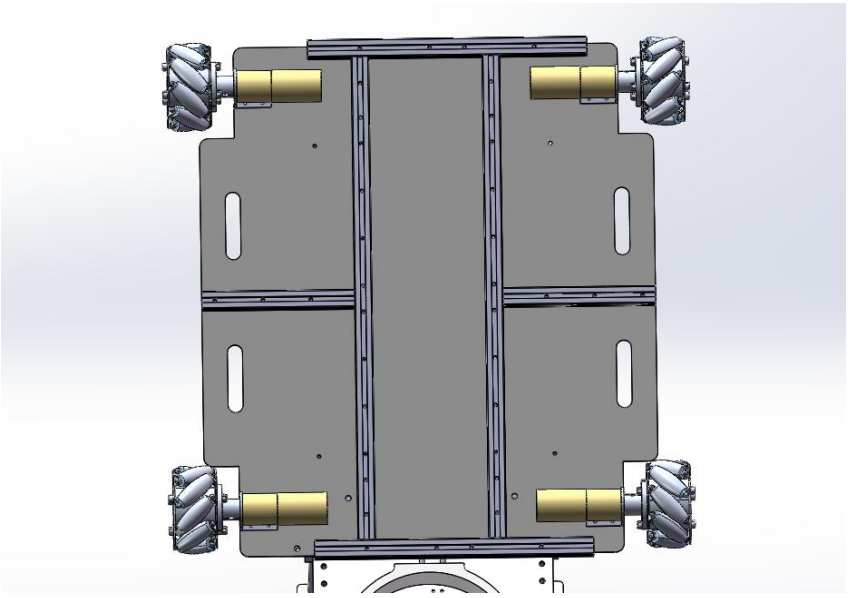
2.2 模块设计与选型

2.2.1 底盘

底盘采用麦克纳姆轮驱动，便于机器人全向移动和转换方向。我们组对麦轮两边都加装了联轴器，使连接件承重平衡，防止歪斜。利用带编码器的直流减速电机来控制麦轮转动。电机功率为 120w，保证有足够驱动力。

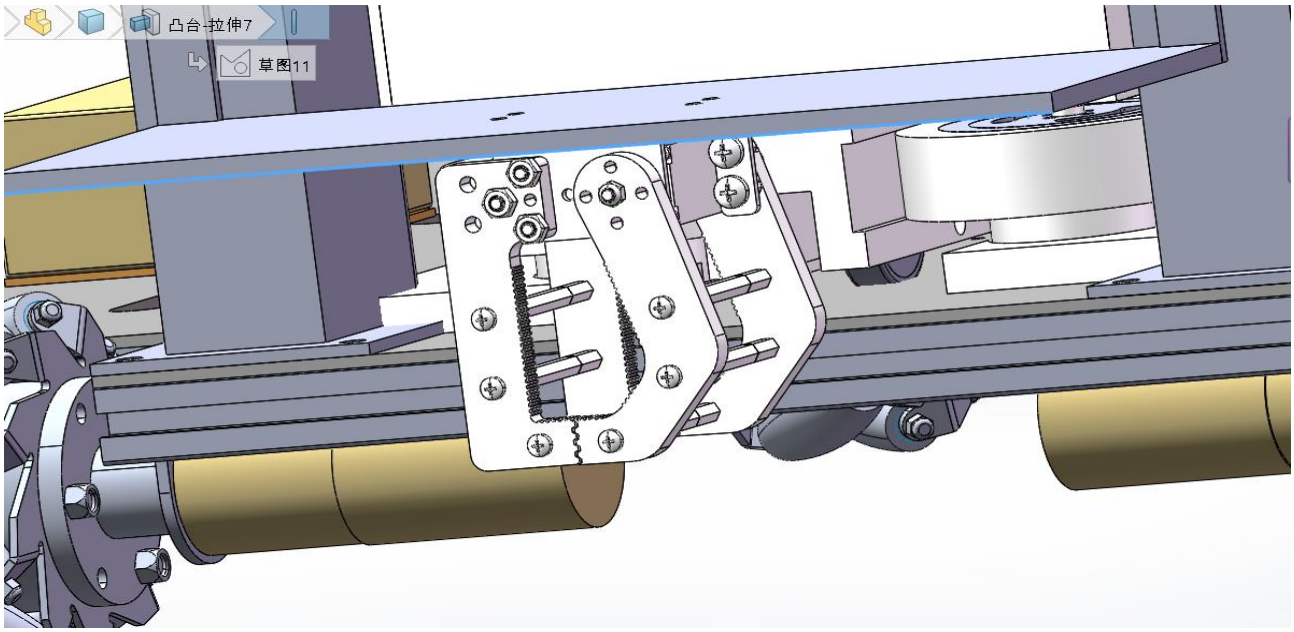


底盘框架采用铝板，用铝型材作为支撑，保证轻便性与稳固性。由于比赛场地较平整，无越障需求，因此未使用悬挂，可保证发射时车体的稳定性。底盘上预留传感器安装点位，同时适当镂空保证轻量化。

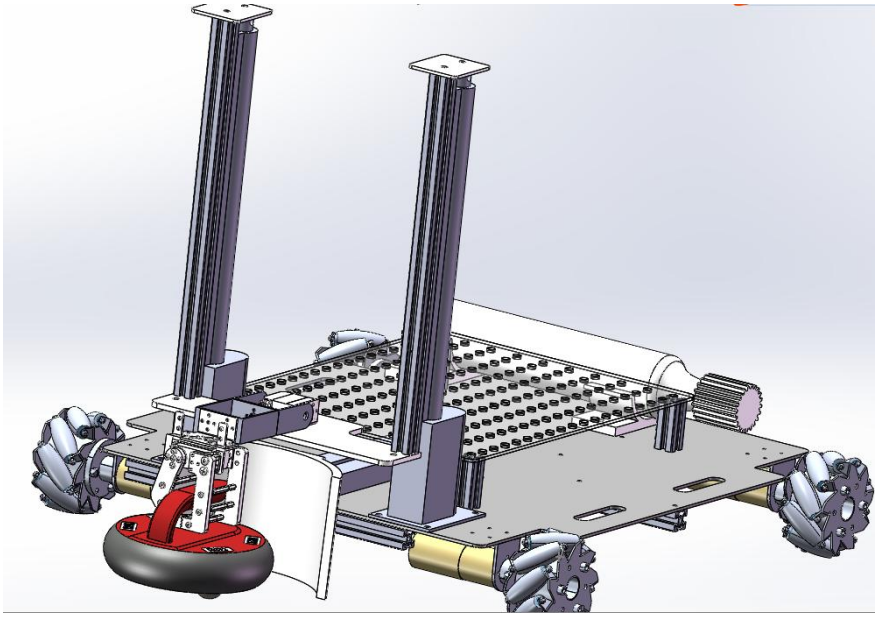


2.2.2 取壶抬升机构

取壶采用竖直机械爪，抓取壶柄，再通过推杆将冰壶抬升至指定高度。机械爪用 20kg 扭力双轴舵机驱动，且爪尖啮合，足以夹起冰壶。



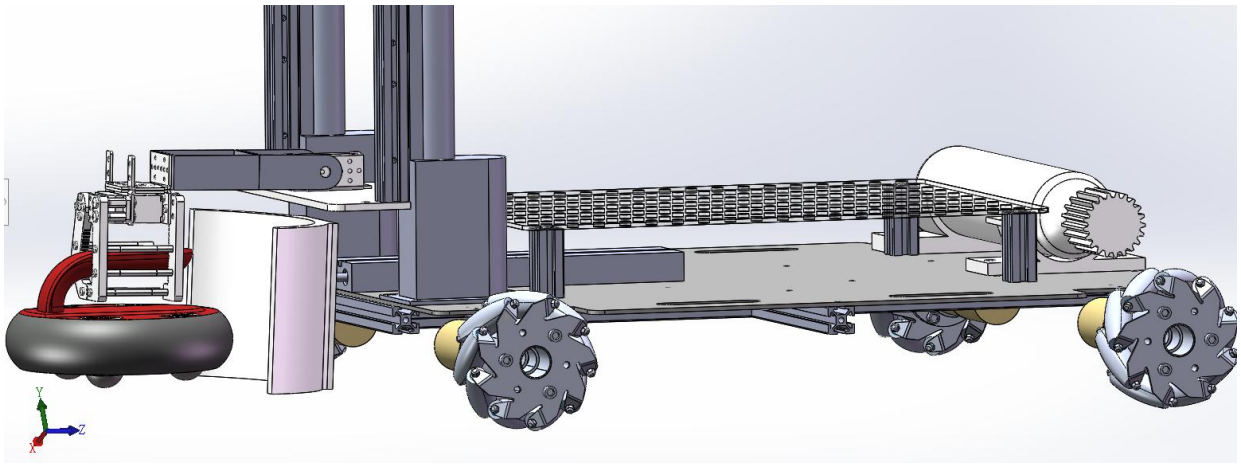
抬升机构借鉴了叉车的运行原理，采用两个电动推杆提供抬升力。电动推杆可伸长 30cm，两个推杆可提供推力达 300N 以上，足以抬升冰壶至指定高度。



2.2.3 发射机构

发射方式：在机器人尾部的气瓶中储存有一定量的高压气体，经过减压阀后降至指定气压，气瓶的开闭可由减压阀控制，最后气体驱动气缸将冰壶推出。

优势：结构简单，且可以通过调节减压阀中气压大小来调节发射力度。



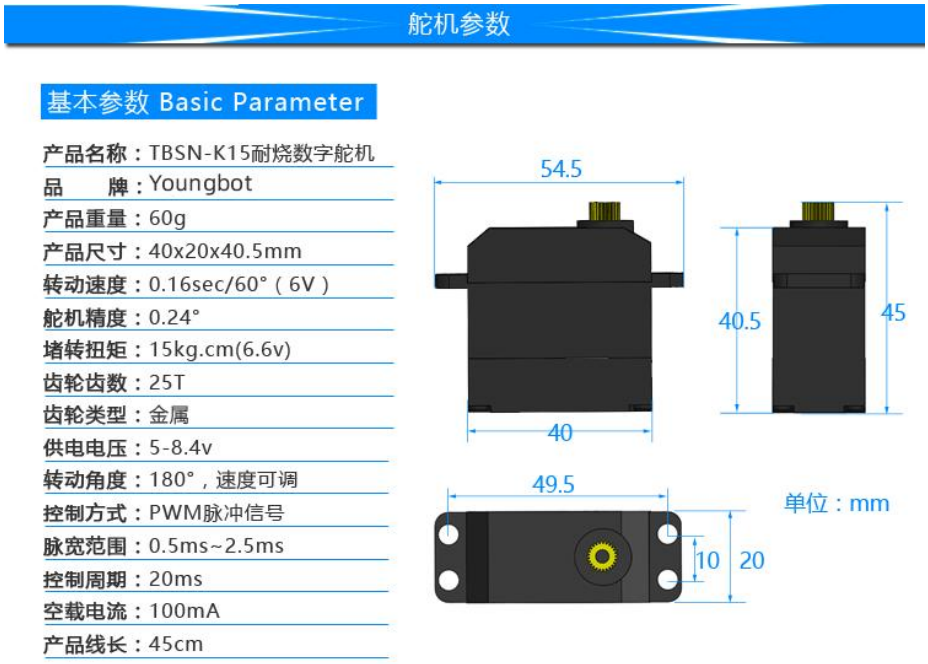
2.2.4 器件选型

麦轮选用如下减速电机：

工作电压 24V，减速比 1:51，最大功率 120W，额定转速为 190 转/分，重量约 480g，带有霍尔编码器。

型号: CHP-36GP-555S 永磁碳刷含霍尔编码减速电机 使用电压: DC24.0V 最大功率120W										
减速比(变比)	1: 3.7	1: 5.2	1: 14	1: 19	1: 27	1: 51	1:100	1:139	1: 264	1: 515
空载电流(mA)	≤600	≤600	≤600	≤600	≤600	≤600	≤600	≤600	≤600	≤600
空载转速 (rpm)	3200	2300	850	630	440	230	120	85	45	23
额定转矩(kg.cm)	3	5	10	15	25	35	50	50	50	60
额定转矩(N.m)	0.3	0.5	1.0	1.5	2.5	3.5	5.0	5.0	5.0	6.0
额定转速 (rpm)	2400	1750	650	500	350	190	90	65	35	18
额定电流(A)	≤2.7	≤2.7	≤2.7	≤2.7	≤2.7	≤2.7	≤2.7	≤2.7	≤2.7	≤2.7
堵转电流(A)	≤21	≤21	≤21	≤21	≤21	≤21	≤21	≤21	≤21	≤21
堵转转力(kg.cm)	≥8.0	≥10.0	≥25.0	≥30.0	≥35.0	≥50.0	≥50.0	≥50.0	≥60.0	≥60.0
减速后编码线数	62.9	88.4	238	323	459	867	1700	2363	4488	8755
减速器长度	26.5mm		34.5mm			42.5mm			50.5mm	

机械爪控制采用 20kg 双轴舵机，精度可达 0.24 度，转动角度达 180°，6.6V 情况下扭矩为 15kg·cm。



电推杆选用 24V，行程 300mm，速度 45mm/S，扭矩 200N，功率约为 20w 的平底电推杆。

单头平底座银色推杆

行程30MM

可选电压 **默认24V** 12V

可选扭矩推速：

速度5MM/S 扭矩1000N
速度10MM/S 扭矩900N
速度15MM/S 扭矩700N
速度20MM/S 扭矩500N
速度30MM/S 扭矩300N
速度45MM/S 扭矩200N
速度65MM/S 扭矩150N
速度90MM/S 扭矩100N

(此图为完全缩回图)

注：推杆电机完全缩回长136.5MM 完全伸出长166.5MM

气瓶选用 0.45L 铝质气瓶，可耐压 30MPa 以上（完全足够推动冰壶）同时选用配套瓶口阀来手动开闭和观察瓶内气压。

参数值

容量（L）	安全压力（MPa）	长*直径（cm）	壁厚（mm）	重量（kg）
0.35	30	25*6.1	6.3	0.80

容量 ：0.45L
安全压力 ：30MPa / 4500psi
测试压力 ：45MPa
爆破压力 ：>80MPa
撕裂压力 ：85MPa
螺纹规格 ：M18*1.5, 每个产品螺牙超过9牙确保绝对安全 .
材质 ：6061 高精铝（航空专用），高轻度、耐疲劳
产品采用500t高压一次成型
壁厚 ：≥6MM, thick explosion-proof.
瓶口高温热收口，杜绝任何焊接
平底标准H型设计，受力均匀安全有保障
应用 ：Grass aquarium aquaculture, medical emergency rescue, mountaineering, soda water machine CO2 supply, fire rescue and other occasions.

参数值

容量（L）	安全压力（MPa）	长*直径（cm）	壁厚（mm）	重量（kg）
0.45	30	29.5*6.1	6.3	0.95

电磁阀选用 24V 电磁阀（注意接口大小）

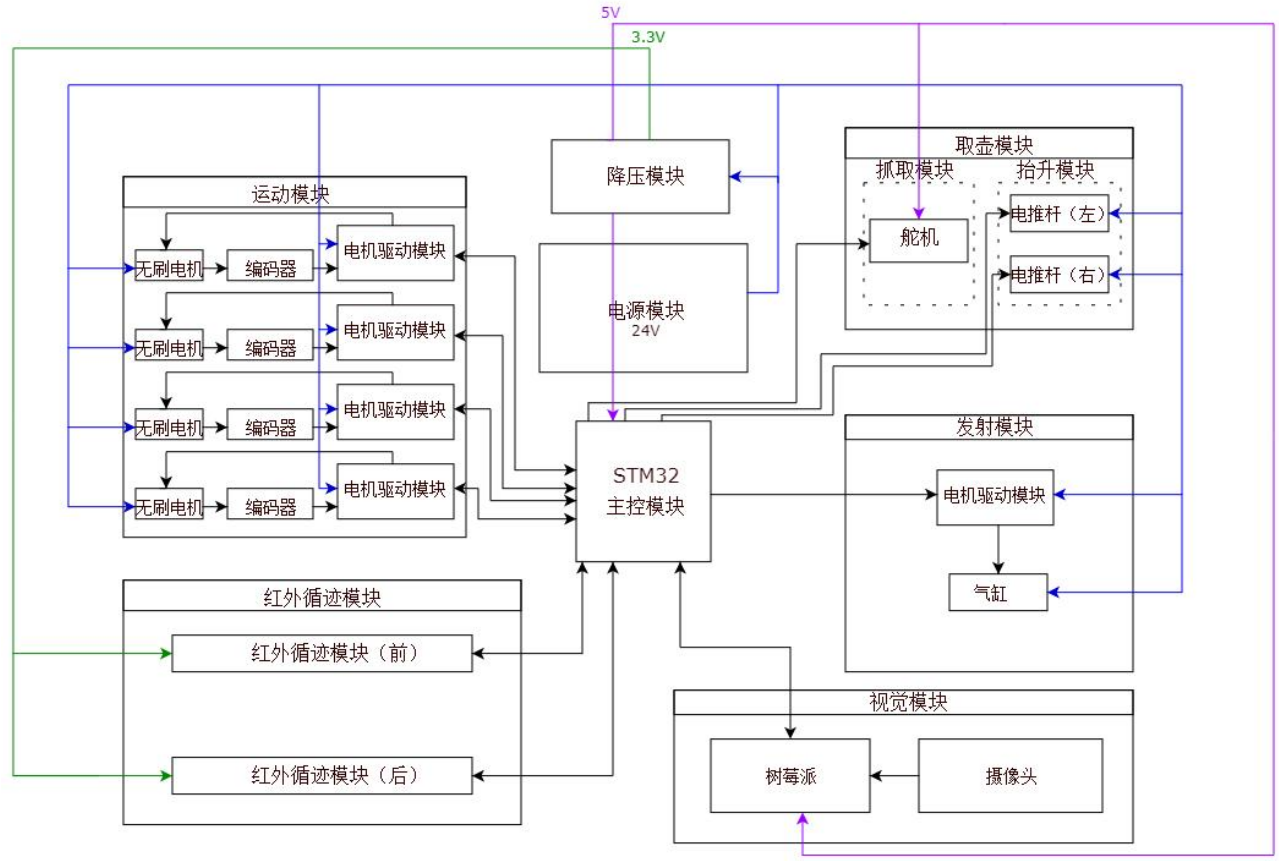
减压阀选用双表减压阀，两表分别显示减压阀两侧气压。

产品型号	公称流量	最高输入气压	输出气压调节范围	输入量程	输出量程	输入端螺纹规格	输出端螺纹规格
YQD-07	40m³/H	15MPa	(0.1-1.5) MPa	25mpa	2.5MPa	G5/8右旋	M16×1.5右旋

气缸选用内径 16mm 行程 200mm 的双轴气缸。

3. 电路部分

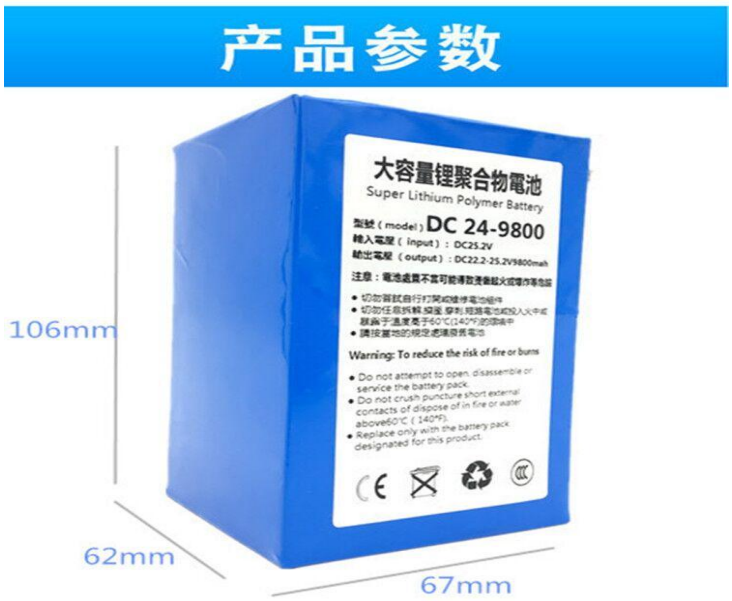
3.1 电路框图



3.2 供电系统

3.2.1 电源

电源采用 24V9800ma 锂聚合物电池，持续电流达 10A，瞬间启动电流达 50A，重量为 861g，实物图及产品参数如下：



型号	24V9800
标称电压	24v
工作电压	16.2V-25.2v
容量	9800ma
电芯	6串A品聚合物锂电芯
尺寸	106*67*62mm
重量	816g
可持续电流	10A
过流保护值	50A

3.2.2 分电方案

名称	数量	电压	电流	总功率
无刷电机	4	24V	≦2.7A	120×4=480W
电动推杆	2	24V		30×2=60W
气缸	1	24V		72×6=422W
舵机	2	5V	≦1.5A	
电机驱动模块	4	3.3V		
STM32 主控模块	1	5V		
树莓派	1	5V		
红外循迹模块	2	5V		

后六种元器件单独消耗的功率很小，电流都以毫安计量，故不纳入计算.

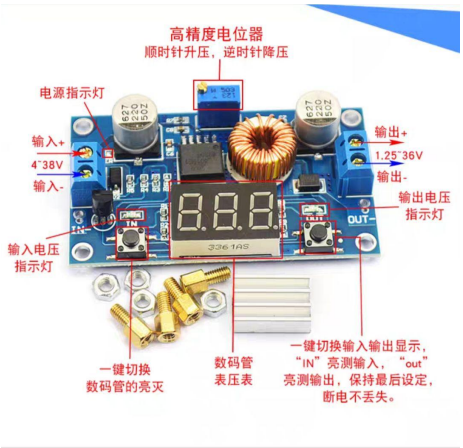
由于本次比赛对取壶速度要求不高，对舵机速度要求较低，相对于摩擦轮和无刷电机的功率可以忽略。

由于无刷电机和摩擦轮不会同时工作，故电源可以满足各元器件对功率和电压的要求。

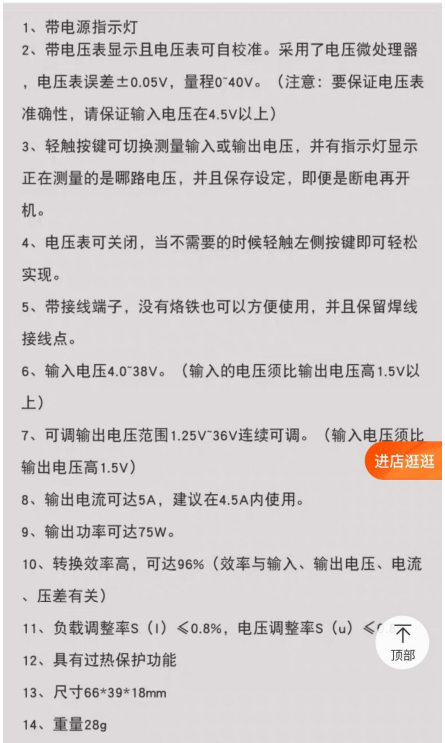
3.2.3 稳压方案

由于电源电压为 24V，而电路中含有输入电压为 14.8V、6.8V、5V 、3.3V 的模块和芯片，故需要使用降压模块来将较高电压转化为合适的电压。由于所需功率较大，且为了方便观察输出电压，所以我们选择以下这款 DC-DC 电源管理模块实现上述功能。

[实物图]：



[功能特性]



根据分析，该款降压模块可调输出范围较大，输出电流较高，可以满足设计中对多种不

同电压的要求，故可以选用该模块作为降压装置。

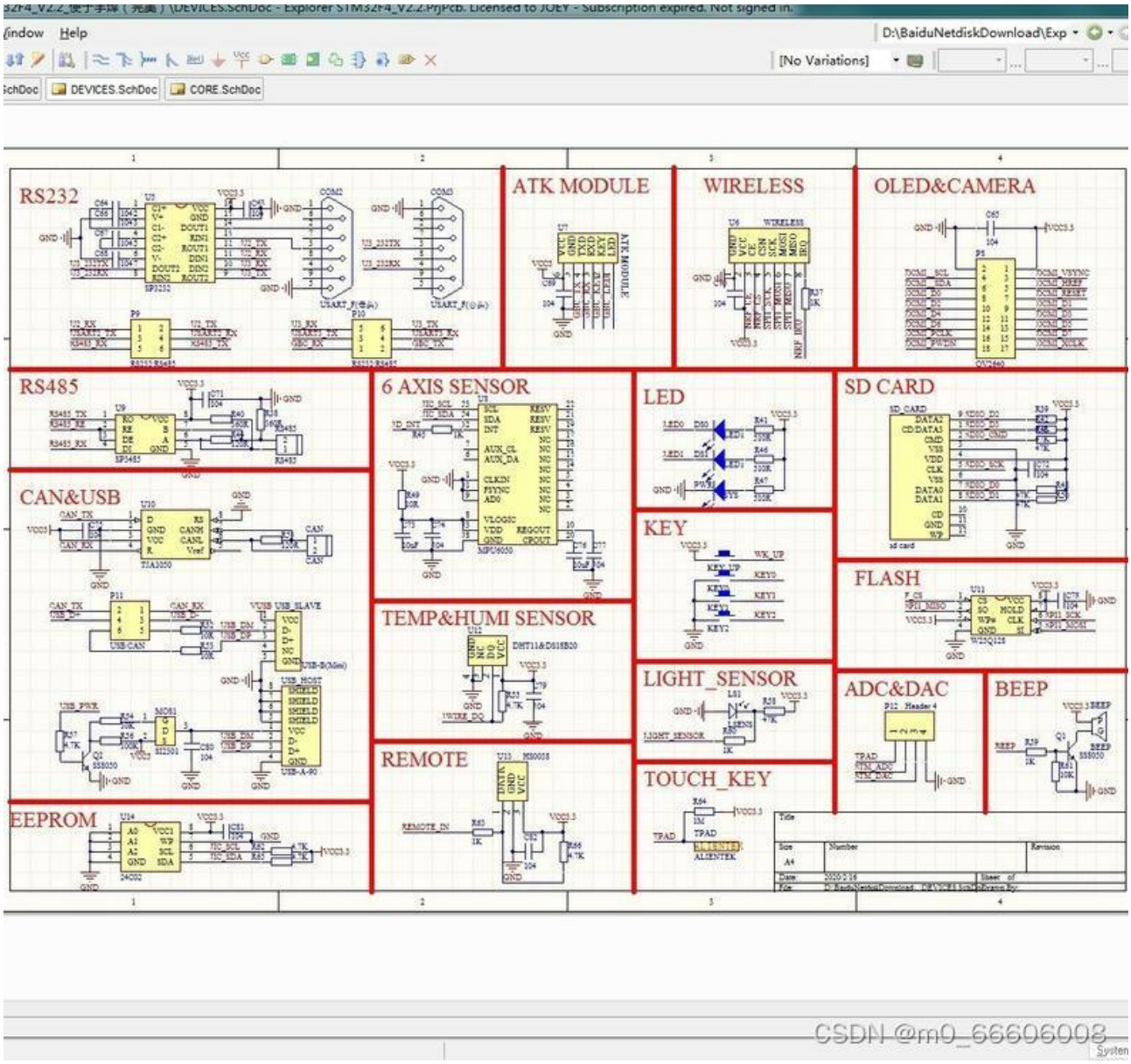
3.3 控制系统

3.3.1 主控模块

根据功能需要，我们选取 STM32F407ZGT6 作为主控模块。

[实物图]

[原理图]



[结构参数]

1、内核：带有 FPU 的 ARM® 32 位 Cortex®-M4CPU、在 Flash 存储器中实现零等待状态运行性能的自适应实时加速器 (ART 加速器 ™)、主频高达 168MHz， MPU，能够实现高达 210 DMIPS/1.25DMIPS/MHz (Dhrystone 2.1)的性能，具有 DSP 指令集。

2、存储器

高达 1 MB Flash

高达 192+4 KB 的 SRAM， 包括 64-KB 的 CCM（内核耦合存储器）数据 RAM

具有高达 32 位数据总线的灵活外部存储控制器： SRAM、 PSRAM、 NOR/NAND 存储器

3、 LCD 并行接口，兼容 8080/6800 模式

4、 时钟、复位和电源管理

1.8 V 到 3.6 V 供电和 I/O

POR、 PDR、 PVD 和 BOR

4 至 26 MHz 晶振

内置经工厂调校的 16 MHz RC 振荡器（ 1% 精度）

带校准功能的 32 kHz RTC 振荡器

内置带校准功能的 32 kHz RC 振荡器

5、低功耗

睡眠、停机和待机模式

VBAT 可为 RTC、 20×32 位备份寄存器 + 可选的 4 KB 备份 SRAM 供电

6、3 个 12 位、2.4 MSPS ADC：多达 24 通道，三重交叉模式下的性能高达 7.2 MSPS

7、2 个 12 位 D/A 转换器

8、通用 DMA：具有 FIFO 和突发支持的 16 路 DMA 控制器

9、多达 17 个定时器： 12 个 16 位定时器，和 2 个频率高达 168 MHz 的 32 位定时器，每个定时器都带有 4 个输入捕获 / 输出比较 /PWM，或脉冲计数器与正交 (增量) 编码器输入

10、调试模式

SWD & JTAG 接口

Cortex-M4 跟踪宏单元 ™

11、多达 140 个具有中断功能的 I/O 端口

高达 136 个快速 I/O，最高 84 MHz

高达 138 个可耐 5 V 的 I/O

12、多达 15 个通信接口

多达 3 个 I2C 接口 (SMBus/PMBus)

高达 4 个 USART/4 个 UART（ 10.5 Mbit/s、ISO7816 接口、 LIN、 IrDA、 调制解调器控制）

高达 3 个 SPI （ 42 Mbits/s）， 2 个具有复用的全双工 I2S， 通过内部音频 PLL 或外部时钟达到 音频级精度

2 个 CAN （ 2.0B 主动）以及 SDIO 接口

13、高级连接功能

具有片上 PHY 的 USB 2.0 全速器件/主机/OTG 控制器

具有专用 DMA、片上全速 PHY 和 ULPI 的 USB 2.0 高速 / 全速器件 / 主机 /OTG 控制器

具有专用 DMA 的 10/100 以太网 MAC：支持 IEEE 1588v2 硬件， MII/RMII

14、8~14 位并行照相机接口：速度高达 54MB/s

15、真随机数发生器

16、 CRC 计算单元

17、RTC：亚秒级精度、硬件日历

18、 96 位唯一 ID

较之 STM32F1/F2 等 Cortex-M3 产品，STM32F4 最大的优势，就是新增了硬件 FPU 单元以及 DSP 指令，同时， STM32F4 的主频也提高了很多，达到 168Mhz（可获得 210DMIPS 的处理能力），这使得 STM32F4 尤其适用于需要浮点运算或 DSP 处理的应用，也被称之为： DSC， 具有非常广泛的应用前景。

考虑到本次比赛实际所需的 GPIO 口数量、通信速度和运算速度需求，这款单片 机足够完成作为主控芯片的任务。

3.4 执行系统

3.4.1 巡线模块

参考组委会在群里提供的 hello world 红外巡线模块并自主设计红外巡线模块。每一块红外阵列板以单片机为核心，8 路 ADC 通道接收 TCRT5000L 光电对管产生的电压值进行强弱分析，进而反映出地面材料灰度，最后通过 IIC 总线传输数据。

TCRT5000 是一种基于红外光学反射原理的传感器，它包含一个红外发光二极管和一个光敏三极管，光敏三极管内部覆盖了用于阻挡可见光的铅。

工作时，TCRT5000 的红外发光二极管不断发射红外线（不可见光）波长为 950nm，当发射的红外线没有被障碍物反射回来或者反射强度不足时，光敏三极管不工作，当红外线的反射强度足够且同时被光敏三极管接收到时，光敏三极管处于工作状态，并提供输出。光敏三极管在工作时其集电极电流值 Ic 约为 1 mA，TCRT5000 红外传感器的工作范围约为 0.2~15mm。

4. 算法部分

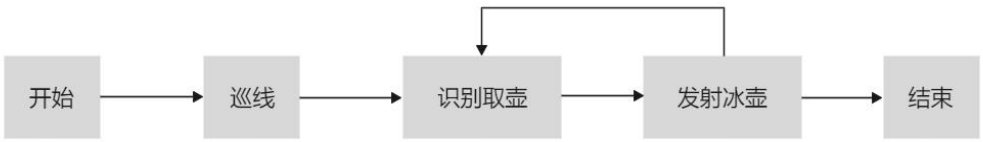
4.1 控制程序架构

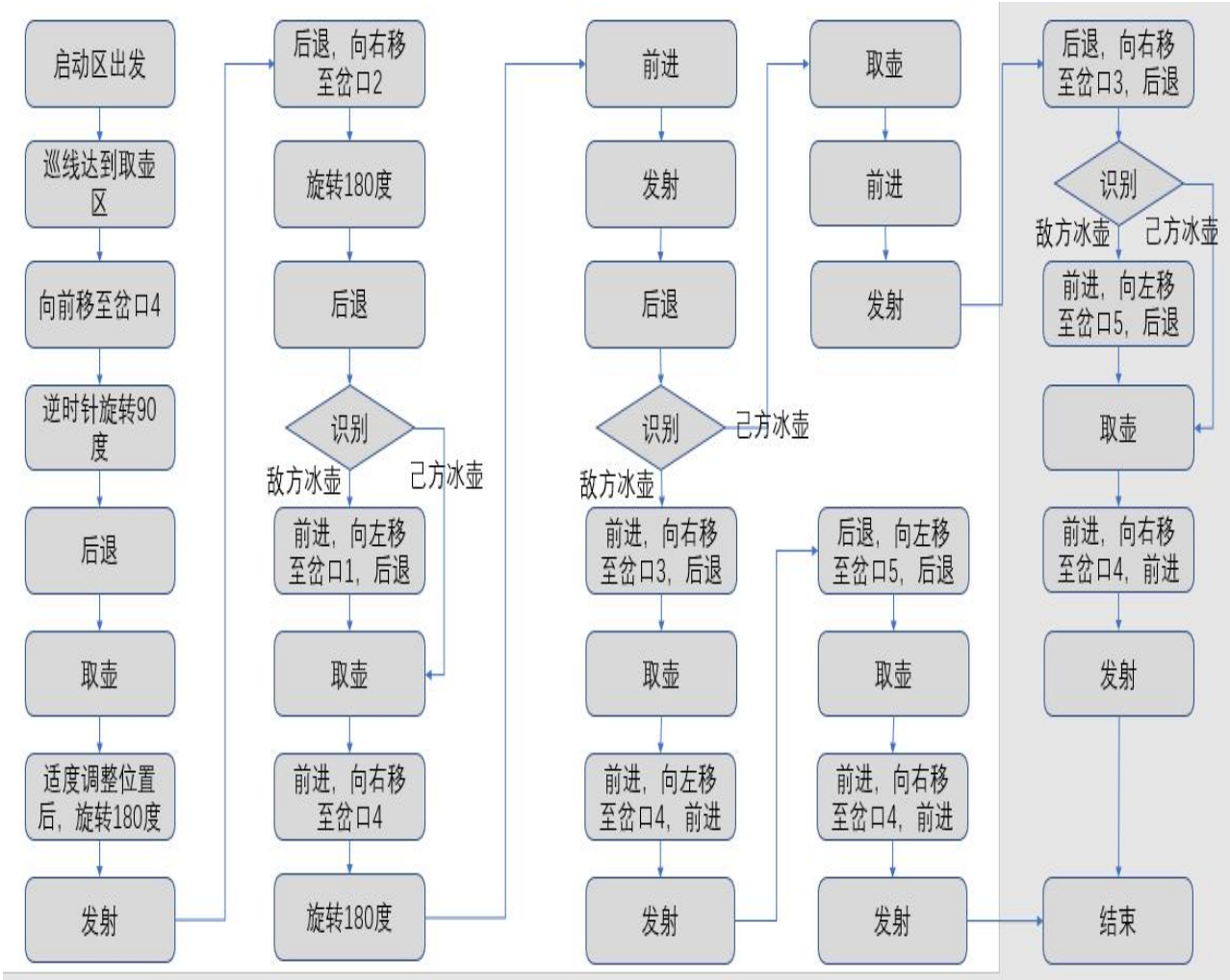
路线	红外巡线	利用红外阵列数据，确定小车的姿态等。
	底盘速度解算	通过霍尔编码器反馈的波的周期，得到电机的转速，计算得速度。
	电机控制	PID 控制电机使转速稳定，PWM 波控制电机速度。
取壶	舵机控制	开环控制，直接使用 PWM 波控制。
	电推杆控制	直接使用直流电控制。
	摩擦轮控制	使用稳压模块输出电压控制。
发射	摩擦轮控制	使用稳压模块输出电压控制。
	舵机控制	开环控制，直接使用 PWM 波控制。

4.2 主控程序设计方案

4.2.1 流程规划

主要流程概述：





4.2.2 控制算法

红外巡迹模块的控制

红外对管中，红外发射管发射一个红外线，经过地面反射后由红外接收管接收。

当黑线经过红外阵列时，不同位置的接收管将接收到不同强度的反射，在黑线正上方的红外对管将接收到反射光少，电阻大，电压大；黑线远处的红外对管将接收到反射光多，电阻小，电压小。

将模拟信号接进单片机的 AD 转换器，将模拟信号转换成数字信号，从而使单片机获取具体的电压值。由此可以得到一条曲线，峰值处即为地面黑线与红外阵列所在边的交点。

两个相对的交点可以确定一条黑线，从而得到机器人相对于黑线的姿态。

霍尔编码器

当马达转动时，会带动一个具有磁性的码盘一起转动，产生一个变化的磁场。霍尔元

件受到磁钢所产生的磁场影响，输出脉冲信号。传感器内置电路对该信号进行放大、整形，输出良好的矩形脉冲信号。

在已知每转一圈产生多少个脉冲信号条件下，将编码器接入单片机定时器模块的编码器接口模式，测量单位时间内的信号的周期数，就可以算出电机旋转的速率。通过相可以确定方向。

PID 算法

我们利用编码器测得电机的转速，将其预设值比较即可得出误差，将其代入公式即可得出调整的幅度。不断改变 PID 中的参数,观察其在不同变化条件下曲线的变化,最终确定合理的数值,即可实现对电机的稳定控制。

这里主要使用增量式 PID 算法。

△u(k) = u(k) - u(k-1)

△u(k) = Kp [e(k) - e(k-1)]+Ki [e(k)]+Kd [e(k) - 2e(k-1) + e(k-2)]

比例 P：当前误差 - 上次误差

积分 I：当前误差

微分 D：当前误差 － 2×上次误差 + 上上次误差

增量式 PID 根据公式可以很好地看出，一旦确定了 Kp、Ti、Td，只要使用前后三次测量值的偏差，即可由公式求出控制增量而得出的控制量。

△u(k)对应的是近几次位置误差的增量，而不是对应与实际位置的偏差，没有误差累加，也就是说，增量式 PID 中不需要累加。控制增量 Δu(k)的确定仅与最近 3 次的采样值有关，容易通过加权处理获得比较好的控制效果，并且在系统发生问题时，增量式不会严重影响系统的工作。

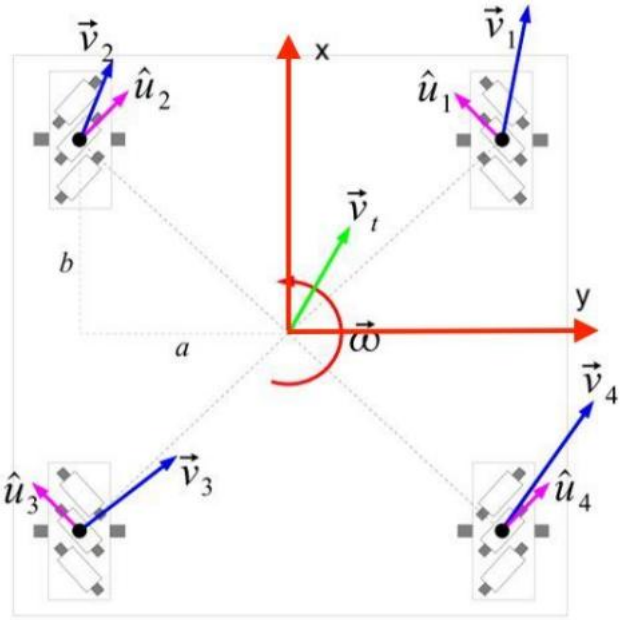
PWM 波控制电机

我们可以做这样的实验，以 24V 直流电机为例，在电机两端接上 24V 的直流电源，电机会以满速转动，如果将 24V 电压降至 2/3 即 16V，那么电机就会以满速的 2/3 转速运转。由此可知，想要调节电机的转速，只需要控制电机两端的电压即可。

所谓 PWM，就是脉冲宽度调制技术，其具有两个很重要的参数：频率和占空比。频率，就是周期的倒数；占空比，就是高电平在一个周期内所占的比例。占空比越大，所得到的平均电压也就越大，电机转速就越快；占空比越小，所得到的平均电压也就越小，电机转速就越慢。STM32 脉冲宽度调制模式可以输出并调节 PWM 波。

麦克纳姆轮的运动解算

机器人的运动状态，可以通过三个量来进行描述 vx，vy，wz，分别表示 X 轴方向上的平动，Y 轴方向的平动，以及围绕 Z 轴旋转。



机器人纵向向前运动时，只需要四个电机以轮毂轴心为圆心，向 X 轴正方向旋转麦克纳姆轮。向后运动时，只需要四个电机以轮毂轴心为圆心，向 X 轴负方向旋转麦克纳姆轮。此时，底盘期望速度只有 X 轴方向的速度。

设 V 是小车的状态量，有如下公式，其中 kx 根据实际情况确定。

V=(vx,0,0)

(v1,v2,v3,v4)=(vx*kx , vx*kx , vx*kx , vx*kx)

机器人需要沿着 Y 轴正方向横向运动时，1、3 号轮沿着轮毂轴向着 X 轴负方向旋转，2、4 号轮沿着轮毂轴向着 X 轴正方向旋转；小车需要沿着 Y 轴负方向运动时，1、3 号轮和 2、4 号轮的旋转方向相反。有如下公式，其中 ky 根据实际情况确定。

V=(0,vy,0)

(v1,v2,v3,v4)=(-vy*ky , vy*ky , -vy*ky , vy*ky)

机器人逆时针旋转要求 1、2 轮向 X 轴正方向旋转，3、4 轮向 X 轴负方向旋转；顺时针旋转要求，1、2 轮和 3、4 轮的旋转方向相反。有如下公式，其中，a 为任意一个麦克纳姆轮到 X 轴的距离，b 为任意一个麦克纳姆轮到 Y 轴的距离。

V=(0,0,wz)

(v1,v2,v3,v4)=(-wz*(a+b) , -wz*(a+b) , wz*(a+b) , wz*(a+b))

仅做好运动模型的分解是不足以达成运动目的的。实际控制过程中，需要电机快速的

响应到期望的转速，这样就要求对每个电机加上单独的 PID 控制。

4.3 视觉方案

4.3.1 视觉实现方式

采取 OpenCV 和单片机（STM32），基于树莓派

我组在本次机器人比赛中采用 OpenCV+Python 的方式实现图像识别和图像处理等一系列功能的实现

4.3.2 OpenCV 简介及图像识别基本想法

OpenCV 中，一个正常的彩色图片由一个三维矩阵组成，每个维度分别为图片的高度、宽度、通道数。

OpenCV 提供两种色彩空间，即 RGB/BGR 色彩空间和 HSV 色彩空间。一般导入的图片是 RGB 空间，而在 OpenCV 提供的算法中多使用 HSV 空间。 RGB 色彩空间包括红色 R、绿色 G 和蓝色 B 三个通道，HSV 色彩空间包括色调 H、饱和度 S 和亮度 V 三个通道。

本次机器人视觉算法中，要求使用算法将正常图片进行二值化处理，生成二值图像，即像素只为 0 或者 255 的单通道图像。这样做的目的是将除目标颜色外的颜色与目标颜色分开，提取掩膜。

为了降低噪声对图像识别的影响，本次算法中将使用腐蚀和膨胀两个方法，平滑二值图像的边缘，提高图像识别的精确度。

由于比赛时会有环境光的影响，图片可能会有一定的颜色偏差，为了使颜色识别更准确，计划使用白平衡算法。白平衡算法有很多，包括均值白平衡算法、完美反射算法、 灰度世界算法、 基于图像分析的偏色检测及颜色校正方法、 动态阈值算法等等。本次机器人视觉算法中计划使用均值白平衡算法。

4.3.3 图像识别基本操作

我们有如下两个方案选择：

方案一：直接对于冰壶进行图像识别，利用 Canny 边缘检测在图像中找到冰壶位置，并进行跟踪；

方案二：较近距离扫描冰壶，通过识别冰壶身上条形码判断冰壶位置和颜色（不用考虑图像中颜色的处理）

考虑到颜色识别的诸多便利性（可以较远分辨出冰壶颜色，减少时间成本；可以较好地

为机器人的行动做出指导），我们优先考虑使用并完善颜色识别方案，条形码识别仅用作辅助。我们采取方案为方案一，下面将进行具体的阐述。

4.3.3.1 基于 OpenCV 的图像识别

我们考虑过深度学习方案，由于它具有学习时间长，代码运算要求高等缺点，不适合搭建在 STM32 单片机上

下面简要介绍我们采取的方式：

安装操作：

在Windows上面安装opencv：设置python的环境变量之后，在命令行输入pip install opencv-python。

```
pip install opencv-python -i https://pypi.mirrors.ustc.edu.cn/simple/
```

树莓派安装: 略

从摄像头里面读取图片：

```
cap = cv2.VideoCapture(0)

    if cap.isOpened() == False:

    cap = cv2.VideoCapture(1)

    ret, frame = cap.read()
```

其中，0和1是摄像头的编号。

图像处理

先将图片缩小至 256×256 大小，使运算时间较短。考虑到有必要降低环境噪声的影响，应当先对图片进行高斯滤波处理。使用 3×3 的高斯核，标准差为 0.8。接下来，考虑到场外地板颜色的影响，以及拍摄到的其他距离较近的目标的影响，需要对边缘具有红、黄、绿颜色的区域进行遮挡。具体的实现方法为：将 BGR 色彩空间的图片转化为 HSV 色彩空间，提取红、黄、绿颜色区域。红、黄、绿在 HSV 空间对应的范围如下表所示。

颜色	H	S	V
红	0-10, 156-180	43-255	46-255

颜色	H	S	V
黄	26-34	43-255	46-255
绿	35-77	43-255	46-255

现在得到了保留红、黄、绿区域的掩模。对该掩膜取反色，将得到遮盖红、黄、绿区域，保留其他区域的掩模。提取该掩模最外层的轮廓，重新绘制轮廓的内部区域，得到的最终掩模将覆盖最外层与图片边缘相连的红、黄、绿区域，而保留内层的区域。对 BGR 空间的图片进行遮挡。之后，将此图片转化为 HSV 空间，提取红、黄、绿区域，并使用 6×6 大小的腐蚀核进行一次腐蚀，以最小化其余冰壶颜色的影响，得到掩模后，对原图片进行遮挡。

4.3.3.2 方案中需要处理的问题以及方法

白平衡问题

由于场地里面有大量环境光影响会对摄像头拍摄和后续图片处理造成很大的影响，所以我们需要进行白平衡优化。

优化方案

```
def white_balance_1(img):
    '''
    第一种简单的求均值白平衡法
    :param img: cv2.imread 读取的图片数据
    :return: 返回的白平衡结果图片数据
    '''
    # 读取图像
    r, g, b = cv2.split(img)
    r_avg = cv2.mean(r)[0]
    g_avg = cv2.mean(g)[0]
    b_avg = cv2.mean(b)[0]
    # 求各个通道所占增益
    k = (r_avg + g_avg + b_avg) / 3
```

```
    kr = k / r_avg
    kg = k / g_avg
    kb = k / b_avg
    r = cv2.addWeighted(src1=r, alpha=kr, src2=0, beta=0, gamma=0)
    g = cv2.addWeighted(src1=g, alpha=kg, src2=0, beta=0, gamma=0)
    b = cv2.addWeighted(src1=b, alpha=kb, src2=0, beta=0, gamma=0)
    balance_img = cv2.merge([b, g, r])
    return balance_img
```

图像处理方式

图像二值化

高斯滤波过滤

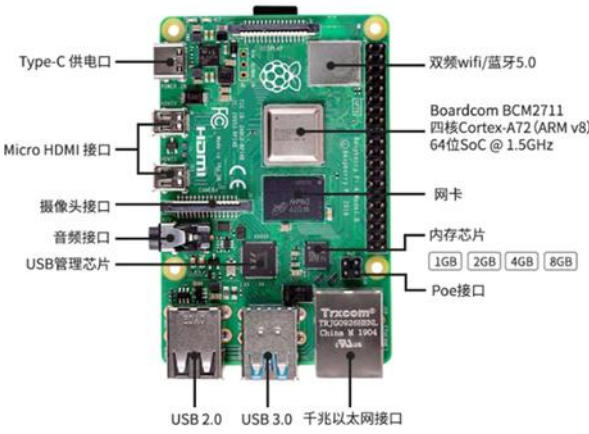
颜色区域提取

腐蚀和膨胀

4.3.4 上位机选择

我们采取树莓派作为我们的上位机。

树莓派 (Raspberry Pi) 是一款为学习计算机编程教育而设计的，基于 Linux系统的微型电脑。相对于面向硬件的 STM32 单片机，树莓派具有更加贴近实际电脑的强大算力，因此在此次机器人比赛中，我们将它作为上位机，进行图像处理操作，并且与 STM32 进行通信，指导机器人的的运动和操作。我们选择树莓派 4b，其实物图和具体参数如下。



型 号	PI3 B	PI3 B+	PI4 B
处理器	64位1.2GHz四核		64位1.5GHz四核
运行内存	1GB		1GB,2GB,4GB,8GB 可根据型号选择
无线WiFi	802.11n 无线 2.4 GHz	802.11n 无线 2.4GHz / 5GHz 双频WiFi	
蓝 牙	蓝牙4.1 BLE	蓝牙4.2 BLE	蓝牙5.0 BLE
以太网网口	100Mbps	300Mbps	千兆以太网
USB 口	4个 USB 2.0 端口		2个 USB 3.0 端口 2个 USB 2.0 端口
GPIO 口	40个GPIO引脚		
视频音频接口	1个全尺寸HDMI端口 MIPI DSI显示端口 MIPI CSI摄像头端口 立体声输出和复合视频端口		2个视频和声音 micro HDMI端口, 最高支持4Kp60, MIPI DSI显示端口 MIPI CSI摄像机端口 立体声音频和复合视频端口
多媒体支持	H.264，MPEG-4解码：1080p30 H.264编码：1080p30 OpenGL ES： 1.1,2.0graphics		H.265：4Kp60解码 H.264：1080p60解码，1080p30编码 OpenGL ES：3.0图形
SD卡支持	Micro SD卡接口		
供电方式	Micro USB		USB type C
POE	无	POE(需额外加模块)	
输入功率	5V 2.5A		5V 3A
分辨率支持	1080分辨率		高达4K分辨率支持双显示屏
工作环境	0-50°C		

树莓派配备使用 USB 摄像头，拍摄清晰度为 720P 高清，内置 CMOS 芯片，图像默认分辨率为 1280×970。该摄像头将拍摄到的图像传给树莓派进行处理使系统获得图像信息。

4.3.5 程序源码及注释

```
import cv2
import numpy as np
import serial

ball_color = 'red'

color_dist = {'red': {'Lower': np.array([0, 60, 60]), 'Upper': np.array([6, 255, 255])},
              'yellow': {'Lower': np.array([11, 43, 46]), 'Upper': np.array([25, 255, 255])},
              }

cap = cv2.VideoCapture(0)
cv2.namedWindow('camera', cv2.WINDOW_AUTOSIZE)

global ser
ser = serial.Serial("/dev/ttyAMA0",115200,timeout=0.5)

read = ser.readall()  # 读取串口传输的数据
if len(read) > 0:  # 接收到内容时转换为 utf8 编码输出（防止乱码）
    print(read.decode('gb2312').encode('utf8'))

flag = 0

while cap.isOpened():
    for i in range(10):
        ret, frame = cap.read()
    frame_after = frame
    if ret:
        if frame is not None:
            gs_frame = cv2.GaussianBlur(frame_after, (5, 5), 0)                    # 高斯模糊
            hsv = cv2.cvtColor(gs_frame, cv2.COLOR_BGR2HSV)                 # 转化成 HSV 图像
            erode_hsv = cv2.erode(hsv, None, iterations=2)                  # 腐蚀 粗的变细
            inRange_hsv = cv2.inRange(erode_hsv, color_dist[ball_color]['Lower'], color_dist[ball_color]['Upper'])
            contours = cv2.findContours(inRange_hsv.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)[-2]
            if len(contours) <= 0:
                c = 0
            else:
                c = max(contours, key=cv2.contourArea)
                rect = cv2.minAreaRect(c)
                box = cv2.boxPoints(rect)
                cv2.drawContours(frame_after, [np.int0(box)], -1, (0, 255, 255), 2)

                area = 0
                for i in contours:
                    area += cv2.contourArea(i)

                print(area)
                if area > 8000.0:
                    flag = 1


            cv2.imshow('camera', frame_after)
            cv2.waitKey(1)
        else:
            print("无画面")

        if flag == 1:
            read = ser.readall()  # 读取串口传输的数据
            read2 = read.decode('gb2312').encode('utf8')
            if len(read2) > 0:
                write = 't'  # 读入要发送的内容
                ser.write("{}".format(write).encode())
            print(read2)
        else:
            read = ser.readall()  # 读取串口传输的数据
            read2 = read.decode('gb2312').encode('utf8')
            if len(read2) > 0:
                write = 'f'  # 读入要发送的内容
                ser.write("{}".format(write).encode())
            # print(read2)
        flag = 0
    else:
```

```
        print("无法读取摄像头！")

cap.release()
cv2.waitKey(0)
cv2.destroyAllWindows()
```