



Using a 55-lb motor driver cable and
Field-Oriented Control (FOC), the
RoboMaster C620 Brushless DC Motor Speed
Controller enables precise control over motor
torque.

Exclusively designed for the RoboMaster
M3AGS P18 Brushless DC Motor and
C620 Brushless DC Motor Speed Controller,
the M3AGS Asynchronous RT includes several
safety and a thermal sensor.

RoboMaster System Specification Manual,
RoboMaster User Manual, Introduction
of RoboMaster System Module

All RoboMaster M3AGS RT includes several
safety and a thermal sensor, ensuring a
complete and reliable system for your
independent robot.

ROBOMASTER

机甲大师超级对抗赛

技术方案

厦门大学 RCS 编制

2024 年 6 月 发布

前言

本技术方案由厦门大学 RCS 编制，适用于 RoboMaster 2024 机甲大师超级对抗赛。

主要撰写人员包括：

模块	撰写人员 1	撰写人员 2
机械	罗聪	江晨峰
硬件	唐佩洁	
软件	陈豪威	
算法	陈豪威	

目录

前言.....	2
1. 概述.....	5
1.1 背景&目标	5
1.2 其他学校机器人分析综述.....	6
1.3 机器人功能定义	6
1.3.1 核心功能取舍.....	6
1.3.2 整体功能定义.....	7
1.4 机器人核心参数	8
1.5 执行器.....	9
1.6 机械臂关键尺寸参数.....	10
2. 设计方案	11
2.1 机械结构设计.....	11
2.1.1 底盘结构设计.....	11
2.1.2 机械臂结构设计	16
2.1.3 重力补偿机构设计	21
2.1.4 存矿机构结构设计	27
2.2 硬件设计	29
2.2.1 整机硬件框图.....	29
2.2.2 硬件详细设计（自研）	31
2.2.3 关键器件选型.....	36
2.2.4 自定义控制器模块	36
2.3 软件设计	41
2.3.1 系统架构.....	41
2.4 算法设计	51
2.4.1 机械臂 M-DH 法逆运动学解算	51
2.4.2 Matlab 仿真	59
3. 研发迭代过程.....	71

3.1 测试记录	71
3.1.1 机械臂	71
3.1.2 取矿测试	74
3.1.3 自定义控制器测试	76
3.2 版本迭代过程记录	77
3.3 重点问题解决记录	78
4. 团队成员贡献	80
5. 参考文献	81
6. 技术方案复盘	82
6.1 赛场性能表现情况分析	82
6.1.1 底盘	82
6.1.2 机械臂	82
6.1.3 自定义控制器	82
6.2 赛场性能表现与规则对比分析	82
6.3 经验总结	82

1. 概述

1.1 背景&目标

本项目是一个面向 2024 赛季 RoboMaster 比赛（后简称 RM 比赛）规则的移动机器人，作为“工程机器人”参赛。在 RM 比赛中，工程机器人的主要任务是搬运场地道具以获得优势，同时进行一些拖拽救援和战术阻挡。工程机器人是辅助型机器人，没有直接进攻能力，但其辅助功能非常重要，能够极大地影响比赛局势。

在 2024 赛季的 RM 比赛规则中，与工程机器人的核心功能非常相关的改动包括经济总量的提升，兑换相关机制的改变，和复活相关机制的改变。

在这里列出几点重要的结论：

- **提升兑换难度对金币数的提升大于提高取矿的数量**
- 半自动工程机器人的收益更高但难度更大
- **视觉兑矿不是决定胜利的方面**
- 工程仍然具有重要的战术阻挡的职能

我们认为工程车的战术职能的优先级为：提供经济>战术作用（阻挡等）。我们的工程车目标具有全向移动，获取和兑换矿石和储存矿石的功能。

综合考虑下，我们决定放弃视觉兑矿。我们在 24 赛季的战术目标是，在 7 分钟的比赛，前 3 分钟内取下并兑换完 3 块金矿，完成后视情况进行银矿拿取和在团战中进行战术阻挡。

1.2 其他学校机器人分析综述

23 赛季，参赛队伍的方案的多样性逐渐发展，主要构型上，大部分队伍采用传统平动平台+末端小机械臂的方案，但采用纯机械臂的学校较之前增加了许多。其中纯机械臂典型的分类可以划分为：

- 6R 型机械臂：上海交通大学、浙江大学、天津大学。
- SCARA 型机械臂：南方科技大学。

其中设计 6R 型机械臂最重要的点是第二个关节的电机需提供的扭矩是最大的。上海交通大学的解决方法是关节电机配合 RV 减速器将扭矩提高，此方式对于没有技术积累的小队伍来讲十分困难；浙江大学的解决方法是将第三个电机向下放置用同步带传动以减小负载，此方法需考虑同步带的张紧和疲劳。

1.3 机器人功能定义

1.3.1 核心功能取舍

1.3.1.1 存矿

如前文所述，我们的目标战术为：前 3 分钟内取下并兑换完 3 块金矿，完成后视情况进行银矿拿取和在团战中进行战术阻挡。

24 赛季的大资源岛改成了通道，这样的通道内取矿会增加取矿时间，而且在大资源岛多次逗留可能会被对面哨兵和步兵截留。这些因素会导致工程取矿回来兑矿一次的时间成本变高。

如果能减少工程机器人在大资源岛-兑换站间移动的次数，可以缩短我们完成所有金矿兑

换的任务的时间以及减少风险。因此我们希望能一次性储存更多的矿石。

因此我们希望存矿机构能**储存 2 个矿石**。这样既能减少在大资源岛逗留的时间，又能一次性拿走小资源岛的三块银矿（存矿 2 个+末端持有一个）。然后到兑换站进行兑换。

1.3.1.2 取矿/兑换

为了确保储矿机构有足够的空间，我们需要尽可能地减少取矿和兑换机构的初始空间。我们可以通过复用取矿和兑换机构、改变其构型以及改变具体机械结构等方式来实现这一目标。在减少空间的同时，机构仍然需要能够稳定地完成取矿、兑换以及与储矿交互等任务。

1.3.2 整体功能定义

功能模块	功能
底盘	<p>能够全向移动</p> <p>上层的机构运动时保持地盘稳定，不移动，不翻车</p>
取矿/兑换机构	<p>能够取到小资源岛矿石</p> <p>能够取到大资源岛矿石</p> <p>能够取到地面矿石</p> <p>5s 内完成一个矿石的获取</p> <p>30s 内完成单次兑换</p> <p>能够完成最高 5 级的兑换</p> <p>工作状态下断电不会砸向地板</p>

存矿机构	<p>能够放入和拿出矿石</p> <p>能够储存两个矿石</p> <p>在机器人的移动和颠簸等过程中，不丢失矿石</p>
其他功能	<p>动态限位</p> <p>各个机构模块化：单个机构 1min 内可以整体从车上拆卸</p>

1.4 机器人核心参数

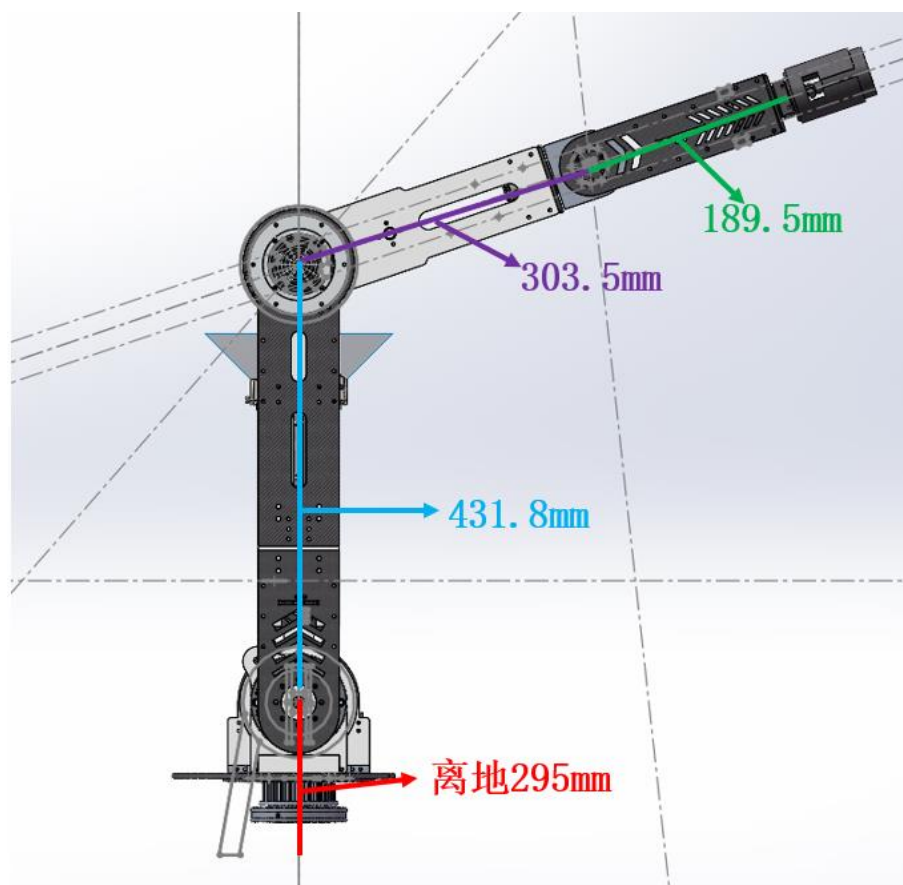
名称	参数
重量、重心	23.2kg
收缩尺寸（长*宽*高）	595*595*598
最大伸展尺寸（长*宽*高）	1100*595*1095
执行器数量	<p>M3508 电机：4</p> <p>达妙 8009 电机：2</p> <p>达妙 8006 电机：1</p> <p>达妙 4310 电机：3</p> <p>真空泵 KMDP-C4：3</p>
最大移动速度	2.5m/s

最大矿石持有量	3
工作真空度	-85KPa

1.5 执行器

名称	用途
M3508	轮组
达妙 8009 电机	机械臂 Joint2、 Joint3 关节
达妙 8006 电机	机械臂 Joint1 关节
达妙 4310 电机	机械臂 Joint4、 Joint5、 Joint6 关节

1.6 机械臂关键尺寸参数



机械臂关键尺寸参数

2. 设计方案

2.1 机械结构设计

在前文的综述中已经提到，尽管存在一些优点，6R 型机械臂工程机器人在设计时需要着重考虑第二个关节。已出现的解决方法有加入 RV 减速器以及下移第三个关节的电机。在此之外，本台工程机器人采用压簧搭配滑轮组进行重力补偿，以减小第二个和第三个关节电机的输出力矩。

本台工程机器人主要由底盘、存矿机构和机械臂三个大模块组成。

零件加工工艺：

零件类型	加工工艺
玻纤板、碳纤板	铣削
非标零件	CNC、钣金折弯、3d 打印
铝方管	钻孔、切割
亚克力板	激光切割

2.1.1 底盘结构设计

2.1.1.1 需求分析

底盘是机器人上层机构的机座，同时机器人提供移动功能。在 24 赛季的规则中，起伏路段并不多，对于我们只取矿石的取矿-兑换的路线来说，地面全是平整的地面以及起伏路段，

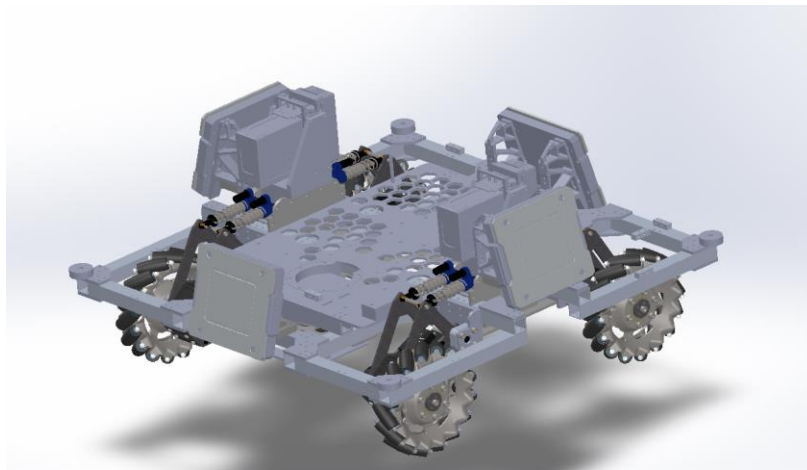
不飞坡，因此对底盘的减震性能要求并没有那么高。同时，在兑矿时最好能有全向移动的功能。除此之外，上层机构（机械臂）运动时，会对底盘产生反作用力，可能造成底盘打滑位移，因此底盘还需要具有对上层机构运动的阻尼减震性能，最好还能相对地面完全锁定。

总结需求如下：

1. 能全向移动
2. 有一定避震性能
3. 相对地面完全锁定（选做）

出于对时间人力的考虑，暂时不实现功能 3，这一版采用普通的麦克纳姆轮移动底盘方案。

2.1.1.2 结构设计



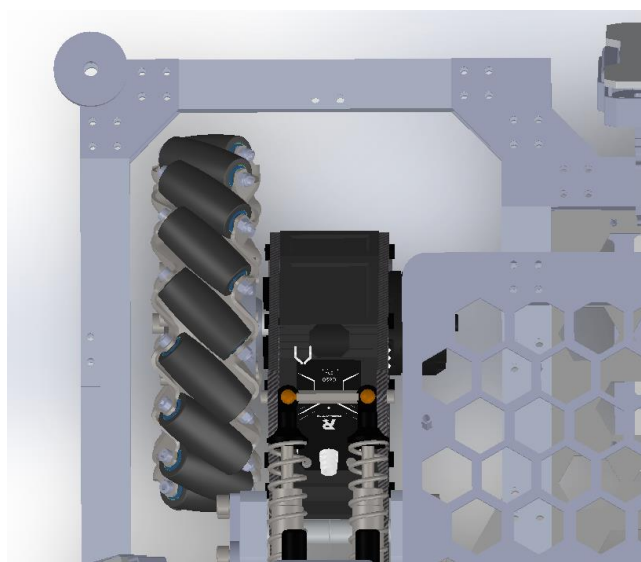
底盘示意图

名称	参数
尺寸（长*宽*高）	580*580*140 (mm^3)
注：高度指框架上平面到地面的距离，不考	

虑超过框架的装甲板等	
下框架离地高度	120mm
重量	13kg
接近角	65.7°
通过角	77.1°
轮距（横*竖）	414*410（mm*mm）

底盘框架主体为 2020 规格的铝方管和 4mm 的玻纤板相铆接，通过玻纤板形状对齐来保证装配精度。

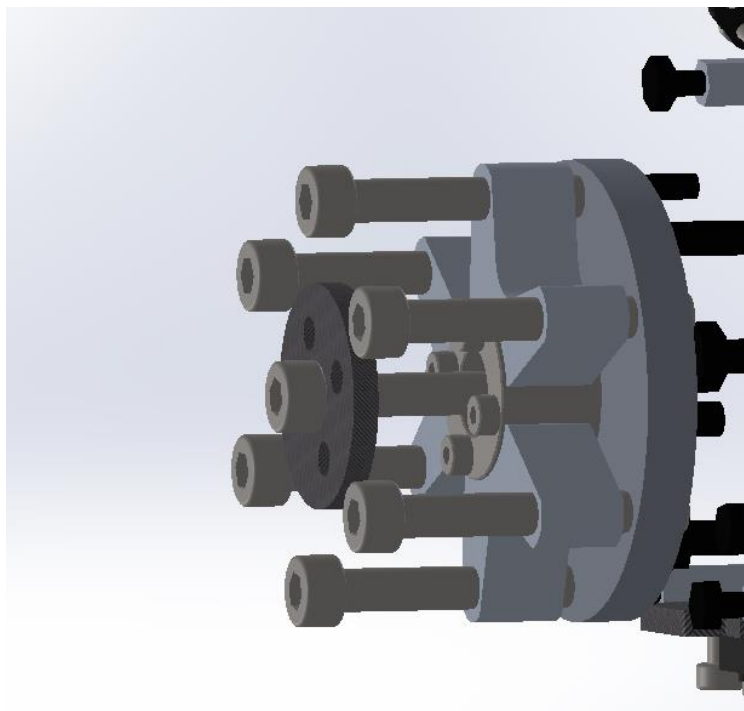
同时这样做会增加底盘质量，工程车的上层机构质量较大，因此底盘质量变高有利于重心的降低，防止翻车。



板材对齐保证装配精度

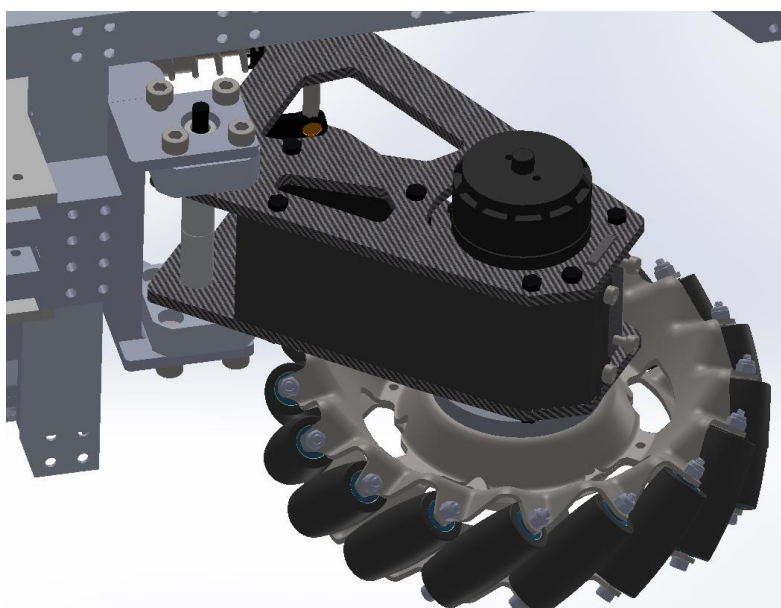
轮组部分，通过机加工件和胀紧套将麦轮锁紧。从左往右分别是：玻纤板小板，机加工

件和胀紧套，机加工件。



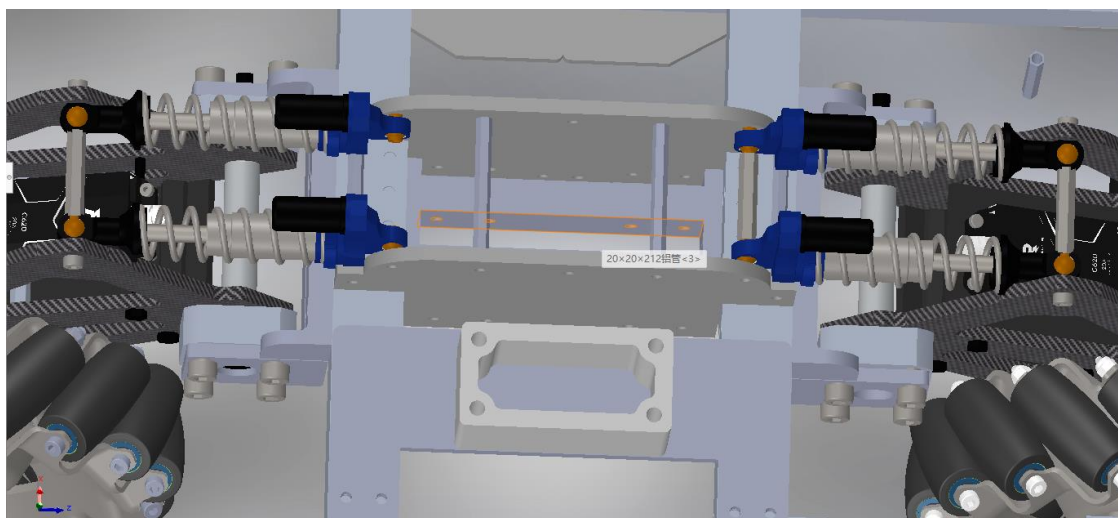
轮组固定示意图

轮组和底盘的固定通过一个机加工件作为支座，中间穿过塞打螺栓和轴套作为加固，整体作为一个模块可以快速拆装，以应对场上的突发情况。



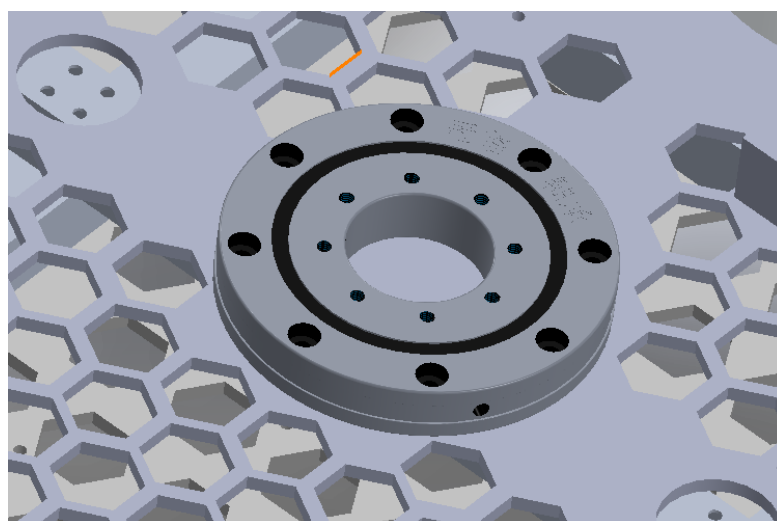
轮组和底盘的连接

而避震器的另一端固定在作为支撑的玻纤板上，两边由玻纤板夹着，中间则是由一根铜柱支撑，这样使得避震器在横向的受力不会导致轮组的横向位移，使避震器仅在竖向上发挥作用，得以发挥其全部性能。而玻纤板和铜柱的强度足够高，并且由于底盘的前后的对称性，纵向的力也能够利用玻纤板的受压相互抵消，从而使底盘具有良好的稳定性。



避震器在玻纤板上的连接

底盘与机械臂的连接则是通过一个交叉滚子轴承来过渡，其上承载机械臂，下与底盘的上板相连，这样使得机械臂也具有模块化的性质，方便拆装。唯一的不足是对轴承的强度要求较高。因此也需要 8 个 $\Phi 3$ 的螺栓来保证稳定性。

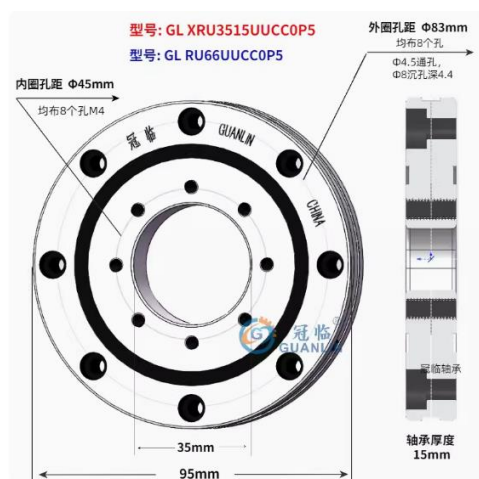


交叉滚子轴承和底盘的固定示意图

2.1.2 机械臂结构设计

2.1.2.1 Joint1

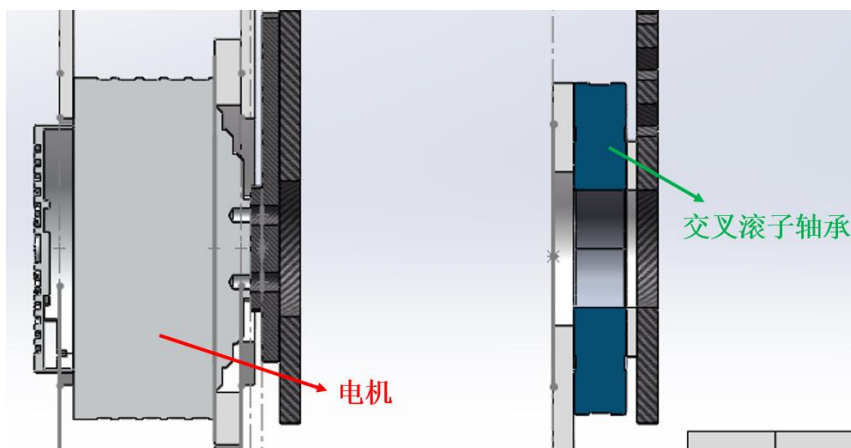
Joint1 转动副采用同步带传动。因为 Joint1 需要承载整个机械臂的重量。并且工作工况恶劣，因此我们直接采用交叉滚子轴承 GL XRU3515UCC0P5，确保能同时承受较大的轴向力和径向力。



交叉滚子轴承 GL XRU3515UCC0P5

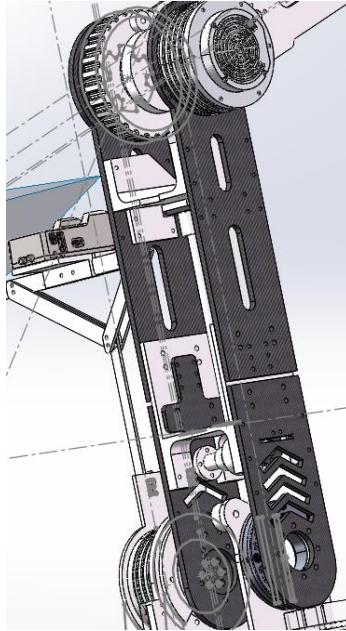
2.1.2.2 Joint2 和 Link2

Joint2 采用达妙 8009 电机直驱，同时也采用交叉滚子轴承 GL XRU3515UCC0P5 同轴从动。这样做的好处同样是，作为承载机械臂重量第二大的关节，重量不是直接作用在电机的输出端，电机只提供扭矩而不是提供支持力。



Joint2 剖面图

为了拆装维修方便，Link2 拆成上下两段。Link 之间的部分被我利用于重力补偿机构的放置，这一部分将在后面进行阐述。



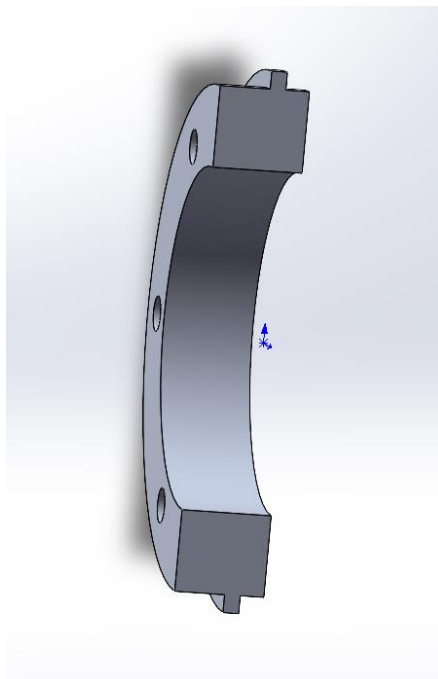
Link2

2.1.2.3 Joint3

Joint3 和 Joint2 设计思路相似：一侧由达妙 8009 电机直驱，另一侧薄壁轴承固定。其中薄壁轴承由两块内有深槽的板子夹住外圈，内圈通过连接件和 Link3 固定。

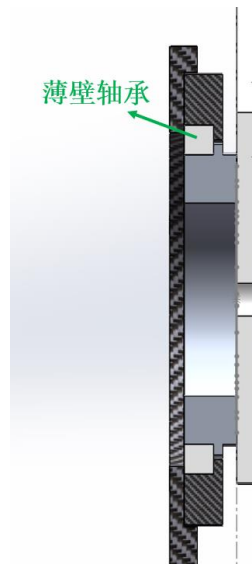


固定外圈的板子



轴承连接件

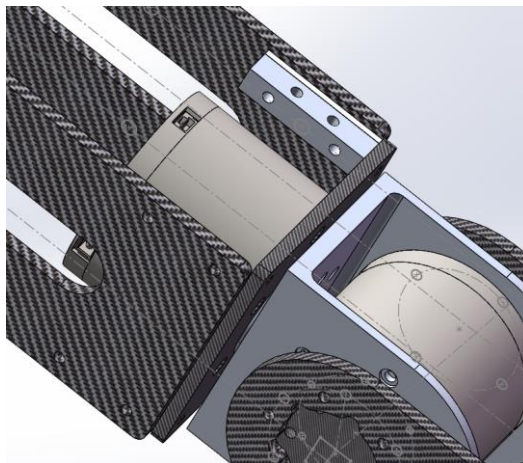
在这个关节使用薄壁轴承而不是之前的交叉滚子轴承是为了减少质量，以减小 Joint2 所要输出的力矩。



轴承固定剖面图

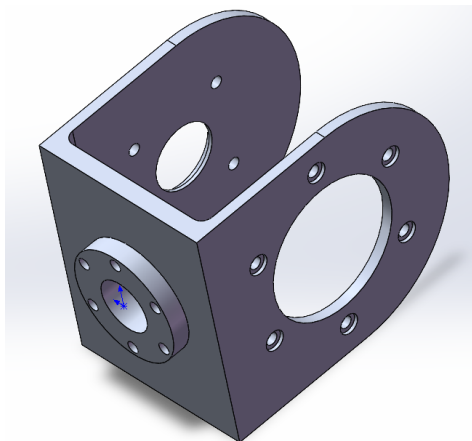
2.1.2.4 Joint4、Joint5、Joint6

末三轴的电机都是用的 4310。其中 Joint4 的固定板是通过六角螺母固定在上一个 Link 上。



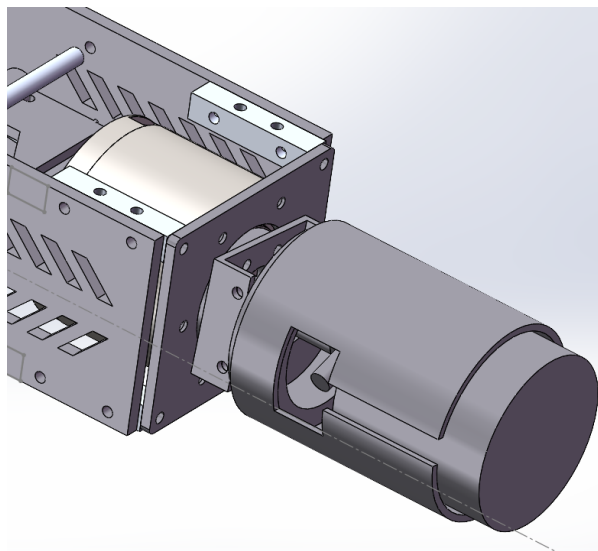
Joint4 固定示意图

为了节省空间，Joint4 与 Joint5 通过一个机加工件连接。



Joint5 固定座

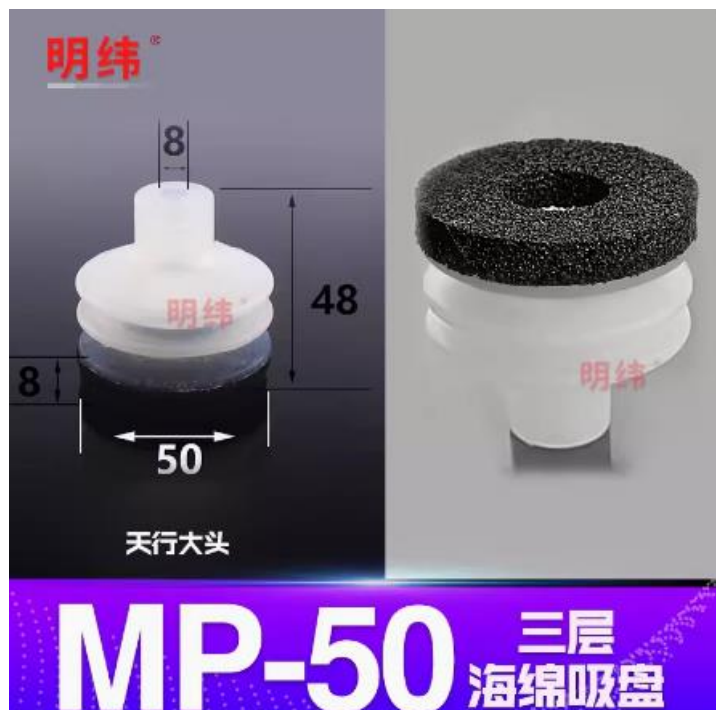
Joint6 的固定方式和 Joint4 相同思路。



Joint6 固定示意图

2.1.2.5 吸盘

在末端执行器的选择上，借鉴上海交通大学的经验，采用 MP-50 海绵吸盘。实际测试中并不会给矿石表面留下印记，也能很好地适应矿石表面的粗糙印花。



MP-50 吸盘

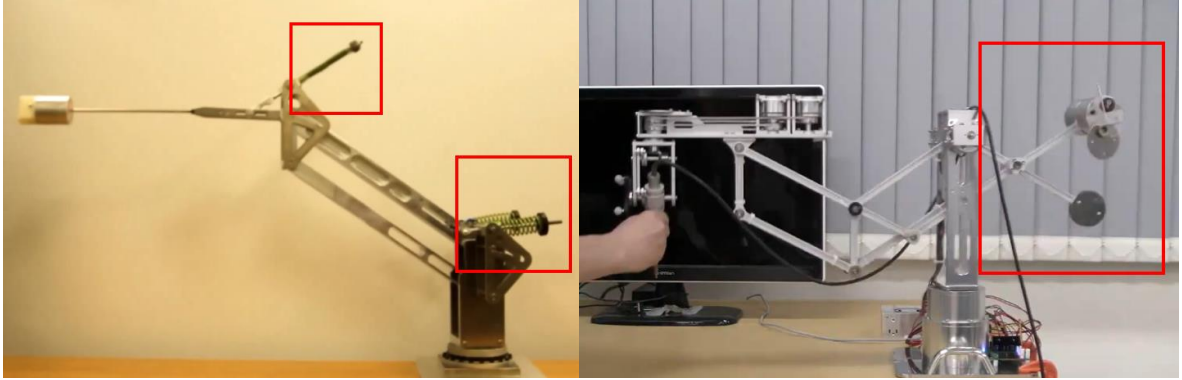
2.1.3 重力补偿机构设计

因为工程机器人的结构通常伴随高自由度（或是抬升行程大），当机器人被击杀时，正在工作的电机会断电，这时候结构猛地砸向地面，会影响本就需要高精度的机器人的性能。除此之外，断电后再次上电时的结构状态是我们无法确定的。这些因素对工程机器人在比赛上的发挥是致命的。此外，前文提到过，6轴机械臂对第二个关节电机的扭矩需求是最大的。在我们的测试中，未加任何补偿的情况下，第二个关节电机的工作扭矩会在 $28\text{N}\cdot\text{M}$ 左右，这样持续工作电机发热明显。综上，我们在本台工程机器人上设计了重力补偿，希望补偿第二、三关节电机的工作扭矩，并且能达到断电后机械臂弹起来而不是掉下去的效果。

2.1.3.1 重力补偿方案对比

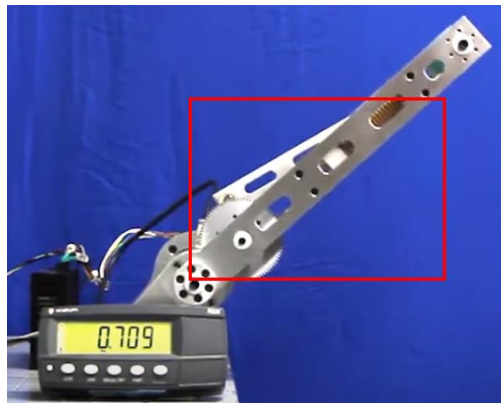
已有的重力补偿方案很多，但是大部分结构会把补偿的结构做在尾部(见下图红框标出)。这样的设计用在工厂车间或是其他需要配备人员的设备上会有安全隐患。而对于工程机器人

而言，本就拥挤的设计空间若加上这样的尾部重力补偿会压缩其他机构的设计。



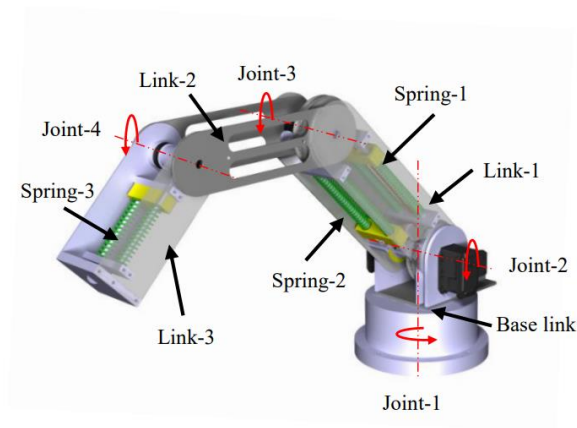
其他尾部重力补偿方案

考虑到机械臂的两个关节之间的 Link 有富足的空间，我们将重力补偿加入到这空间里。下图所展示的是在连杆中的空间采用曲柄摇杆机构压缩压簧达到补偿扭矩。这样的设计虽然没有在尾部挂上那么多的结构，但是还是没有将整个结构放在连杆内。在下图的视角可以看到，还是有一部分结构是裸露在外侧的。



连杆中的重力补偿方案

查阅资料，发现下图的思路可以将第二、三轴关节电机的重力补偿结构放在同一块连杆空间内。自此，我们的工程机器人重力补偿方案确定采用这种思路。

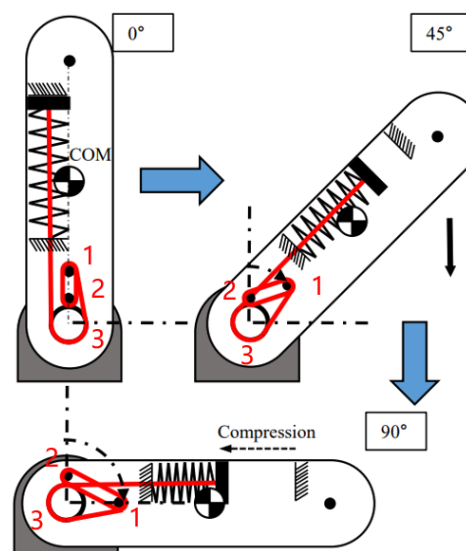


选定的重力补偿方案

2.1.3.2 第二轴关节电机补偿

原理说明

重力补偿的原理是一种动滑轮组的应用。为了方便理解，在下图中标出 1、2、3 滑轮，红线是连接的总长一定的钢丝绳。其中，滑轮 2 和滑轮 3 都是定滑轮。滑轮 1 固定在机械臂上，跟随着机械臂运动。



重力补偿原理

当机械臂从 0° 下摆，滑轮 1 和滑轮 2 的间距增加，由于钢丝绳总长一定，滑轮 1 和滑轮 3 增加的长度便要从连接着压簧的部分补偿。所以当机械臂下摆过程中压簧逐渐压缩。压簧

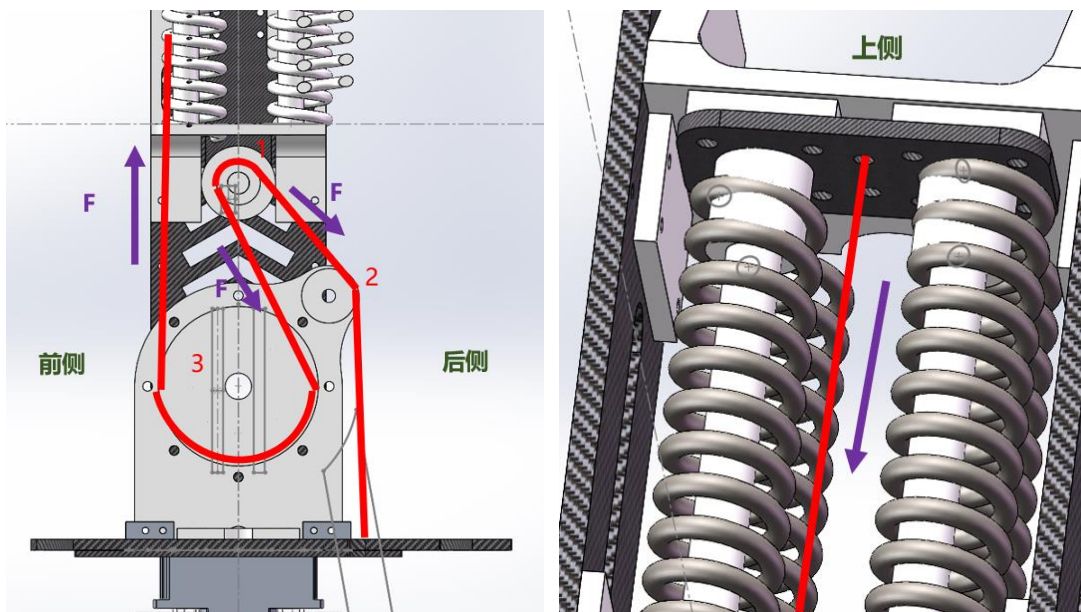
压缩就会有有沿着钢丝绳的拉力。

实际改进

下图是根据以上所讲的原理进行改进的方案。

其中，改变了钢丝绳的绕线方向。这样的绕法两个 F 力都有在垂直于 Link 的方向有分力，都有贡献扭矩补偿。具体的计算会在后文提及。

改变了滑轮 2 的位置。竖直状态下，滑轮 2 不再是在滑轮 1 和滑轮 3 的连线上，而是偏置到一侧。因为工程机器人需要摆动到负的度数（竖直是零度，向前摆动是正的度数），如果不做偏置容易发生钢丝绳从滑轮槽内滑出，结构失效。所以偏置的滑轮 2 起到张紧钢丝绳的作用。



实际采用的重力补偿方案

压簧选型

模具弹簧能够制导的准确 k 值，因此我们使用模具弹簧作为该机构的压簧。


所选用的弹簧型号为 SWF25-200，计算可得 k 值为 4.9N/mm 。采用两个弹簧并联，所以

总的 k 值为 9.8N/mm。

极轻负荷		外径	内径	最大 压缩量	压缩到50% 产生的力		长度
颜色: 黄色		mm	mm	mm	N	kgf	mm
材质: 合金钢		6	3+	50%	59	6	10-300
代号: SWF或YSWF		8	4+	50%	79	8	10-300
工作温度: 200度以内		10	5+	50%	98	10	10-300
		12	6+	50%	137	14	10-300
		14	7+	50%	177	18	15-300
		16	8+	50%	206	21	15-300
		18	9+	50%	255	26	20-300
		20	10+	50%	314	32	20-300
		22	11+	50%	392	40	20-300
		25	12.5+	50%	490	50	20-300
		27	13.5+	50%	588	60	25-300
		30	15+	50%	706	72	25-300
		35	17.5+	50%	960	98	30-300
		40	20+	50%	1254	128	40-500
		50	25+	50%	1960	200	50-500
		60	30+	50%	2820	288	60-500
		70	40+	50%	3920	400	70-500

标准长度: 10、15、20、25、30、35、40、45、50、55、60、65、70、75、80、85、90、95、100、125、150、175、200、225、250、275、300、350、400、450、500。单位:毫米 (mm)

弹簧每毫米力度计算公式 =K值 (弹簧常数) 任何长度都能以此计算
公式: $K = \text{最大力度} \div (\text{自由长} \times \text{压缩率})$ 单位: N/mm (kgf/mm)
举例: SWF25-100, 50kgf $\div (100 \times 0.5) = 1\text{kgf/mm}$



模具弹簧

经测试，机械臂从 0° （竖直状态）下摆至 80° ，弹簧行程为 65mm。两根拉在滑轮 2 上的钢丝绳与 Link 夹角分别为 18° 和 26° 。可以算出每增加一度，压簧会多提供 5.951N。

$$\frac{\frac{9.8N}{mm} * 65mm * (\sin 18^\circ + \sin 26^\circ)}{80^\circ} = 5.951N/^\circ$$

滑轮 2 距离第二个关节的轴距为 93mm，则每增加一度，压簧会多提供 $0.553N \cdot m$ 的扭矩。压到 80° 时。压簧将提供：

$$0.553N \cdot m * 80 = 44.24N \cdot m$$

经测量，机械臂总重 12.7kg (124.46N)，质心离轴距离为 350mm，当压到 80° 时，所需扭矩应为：

$$124.46N * 0.35m * \sin 80^\circ = 42.9N \cdot m$$

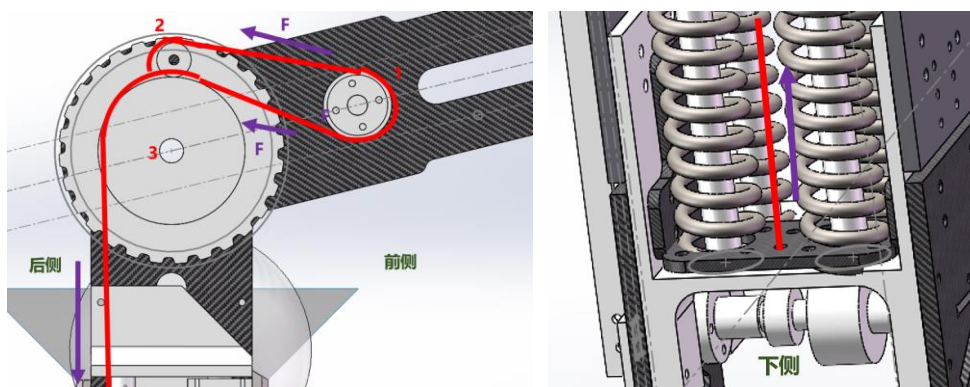
可以看到在机械臂下摆到 80° （本台机器人限位到 80° ）压簧提供的扭矩完全能补偿机械臂的重力。

方案反思

机械臂的重力矩并不是一个线性变化的数，像压簧这样以一个输出线性力的结构很难做到与重力矩完美地拟合；并且机械臂末端姿态不是固定的，我们上面的计算是按照机械臂全部伸展成一条直线的姿态计算，所以剩下的扭矩差值需要电机来提供。也就是说想要做到断电能定在任意位置的效果很难实现。但是组装起来多次测试，8009 电机最大的输出扭矩不会超过 $10\text{N}\cdot\text{m}$ ，电机发烫现象消失。之前预想的断电后机械臂弹回竖直状态的效果是能实现的。

2.1.3.3 第三轴关节电机补偿

第三轴关节电机补偿的原理和第二轴的相同。下图中标出的三个滑轮编号作用和 2.1.3.2 中相同。不一样的是第二轴的压簧是向下压缩的，第三轴的是向上压缩的。

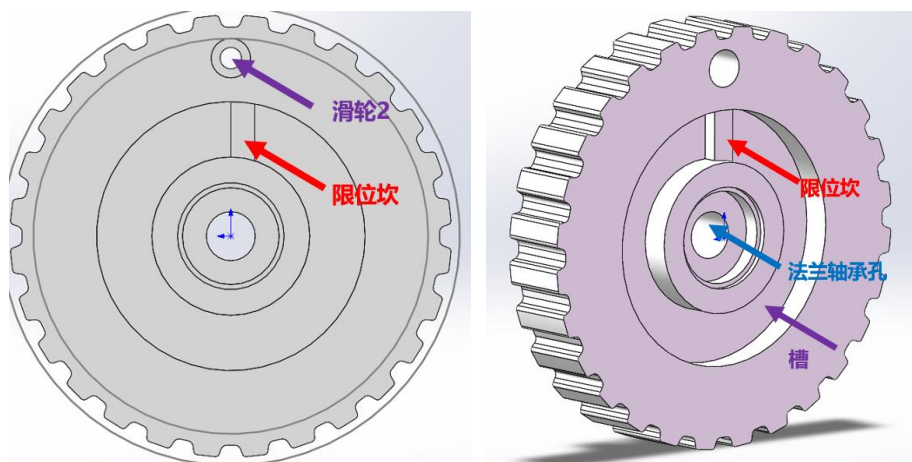


第三轴重力补偿方案

这个补偿方案最重要的一点是滑轮 2 固定的，在本机器人上滑轮 2 和滑轮 3 的连线需保证始终竖直。具体解决方法是在第二轴安装一个同轴的固定的同步带轮 1，而在第三轴也安装一个与第三轴同轴的可转动的同步带轮 2，第三轴的滑轮 2 固定在同步带轮 2 上，这样机械臂第二轴的摆动角度就可以通过机械结构传递到第三轴上。

此外，第三轴的机械限位也是由同步带传递的信息来设计。下图中标注出的槽是电机输出端螺丝的转动范围，槽的深度有 8mm ，限位坎突出了 4mm 。电机 8009 输出端的 8 个固定

螺丝突出 4mm。用螺母垫高 3mm 8 个固定螺丝中的一个，这样这一颗螺丝就不会被允许通过限位坎。在机械臂运动时，限位坎和滑轮 2 总是竖直的，这样就保证了 Link3 的运动范围。



同步带轮 2 示意图

2.1.4 存矿机构结构设计

2.1.4.1 需求分析

储矿机构是我们今年完成战术目标的核心机构。需求如下：

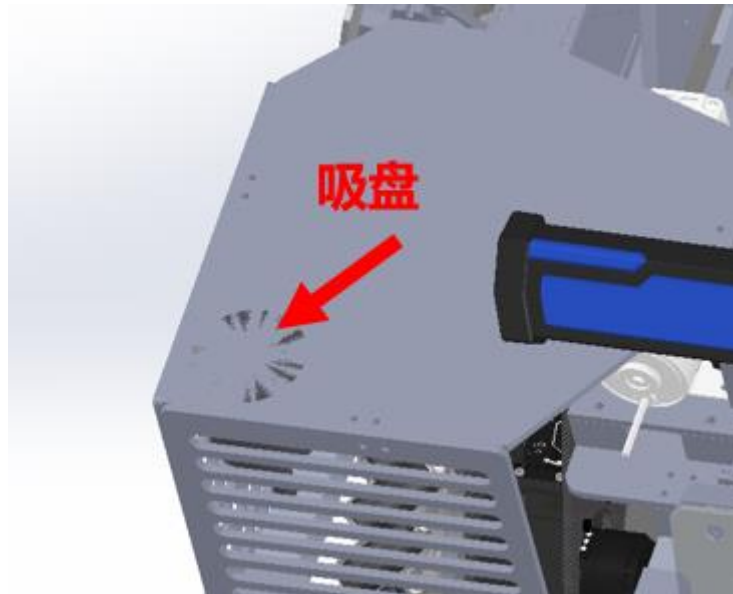
- 1.能储存 2 个矿，加上吸盘自己带一个可以总共完成 3 个矿的同时储存与运输；
- 2.在一固定位置放入矿石，在一固定位置（不一定与放入位置相同）取出矿石；
- 3.整体结构稳定安全，不会出现因撞击等丢失矿石或卡住的情况。

4.金银矿石都是大小为 $200 \times 200 \times 200$ 正方体。工程车有最大初始尺寸限制。尺寸符合工程车的要求，不干涉装甲板，且给机械臂留出必要空间；

2.1.4.2 结构设计

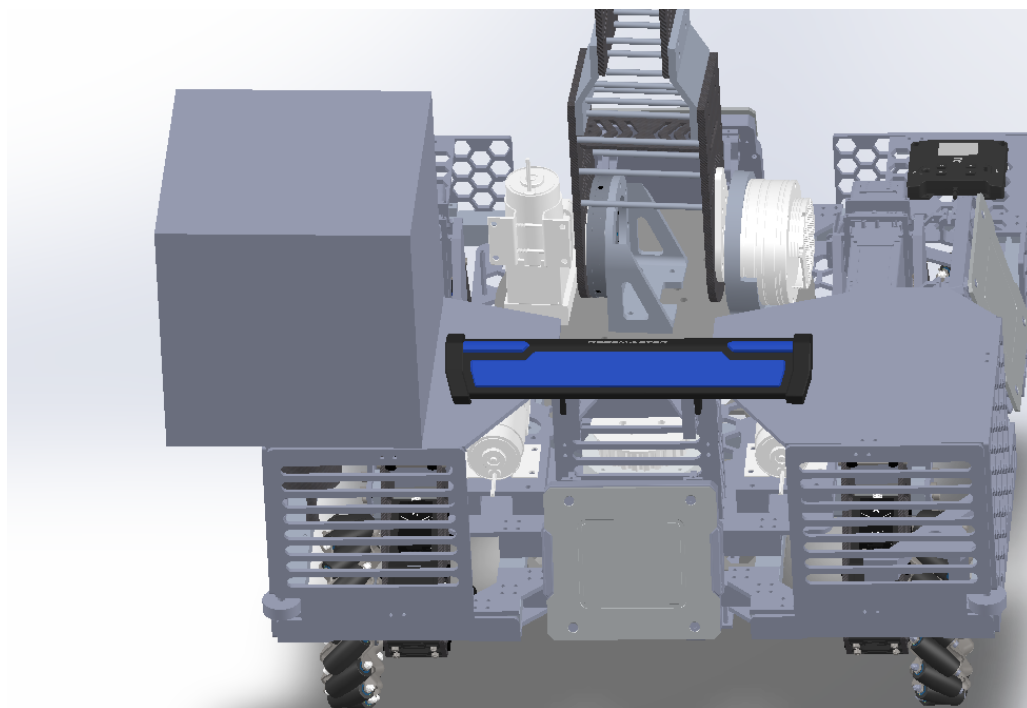
为了方便的取出及存放矿，我们需要保证每次都存矿和取矿都在相对固定的位置；但是为了操作手的操作方便以及对于需要对每次操作误差的容错率保证，我们没有选择对矿石本

身添加过多的限制，而是采用吸盘的方式来存矿，通过固定吸盘，相对放宽矿石放置位置的方法来增加容错率和稳定性。



存矿处吸盘位置

由于吸盘本身的对于气管通路所要求的空间以及吸盘的特性，选择将吸盘放置在整车的前方两角上，这样就可以使得在不超出尺寸限制的情况下最大化的利用空间来存矿，将部分矿石在存放时超出车身，同时有空间来走吸盘的气路。

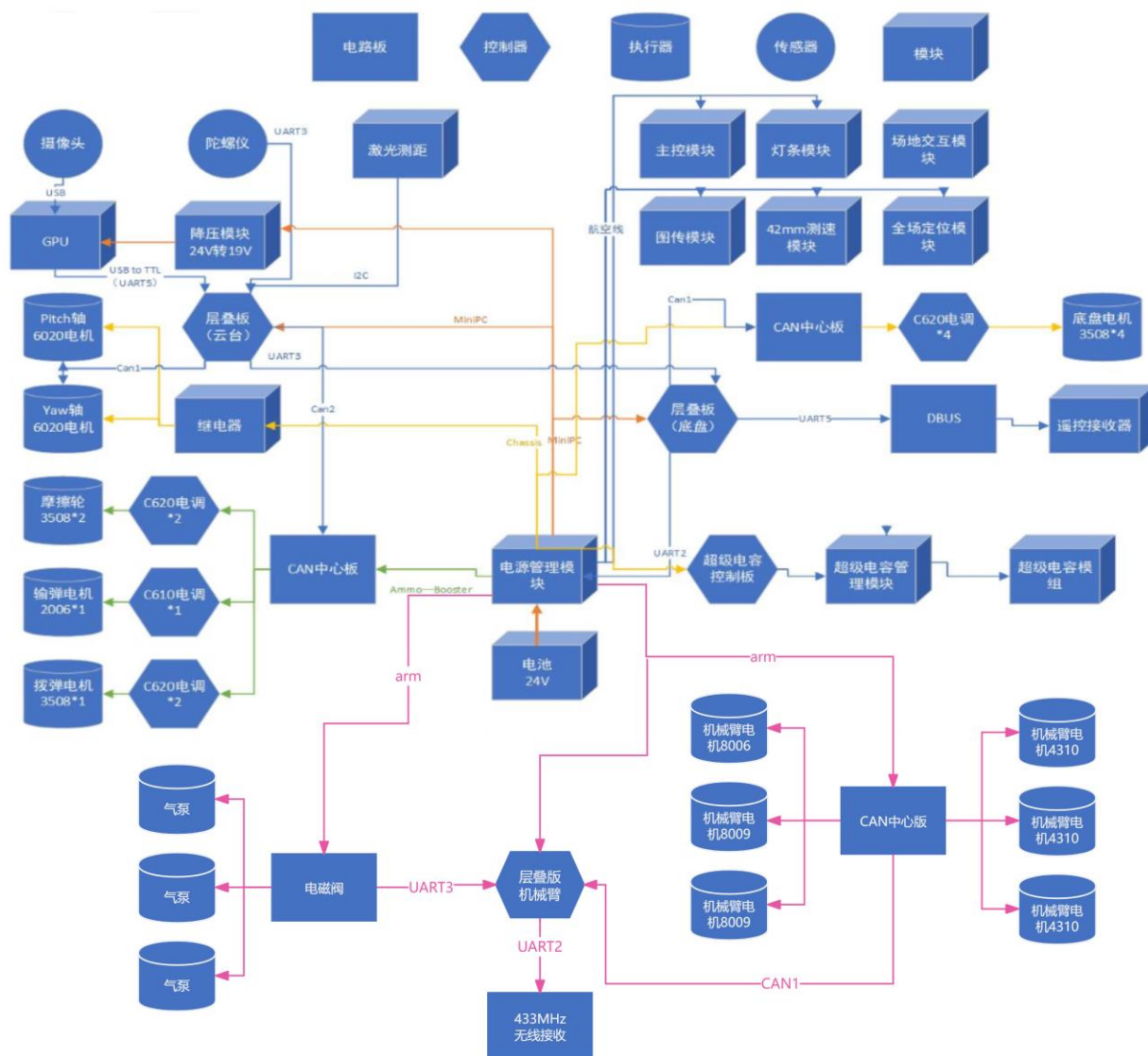


存矿后大致的效果图

2.2 硬件设计

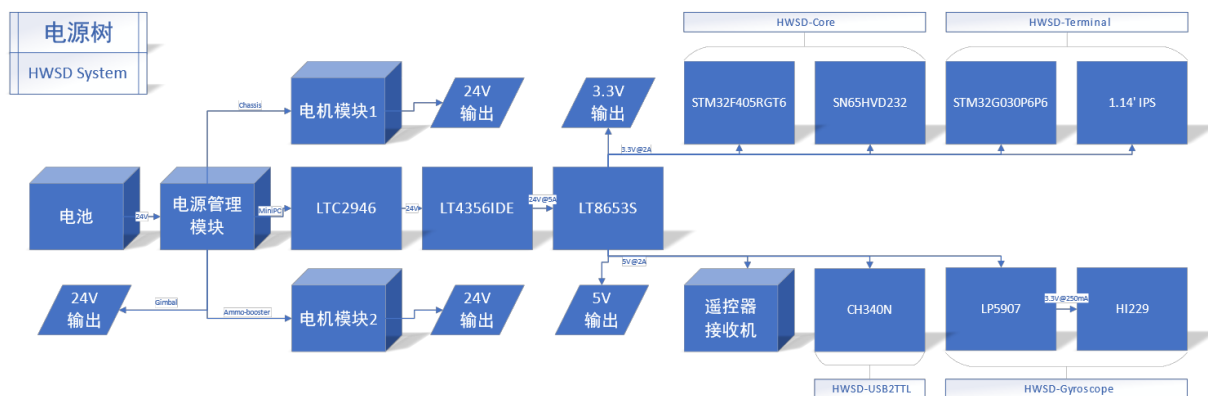
2.2.1 整机硬件框图

本机器人执行器使用 CAN 通信进行控制，传感器使用 UART 串口通信协议交换数据。各块电路板按照硬件标准化的规范进行设计，组装成电源模块、控制模块、电机模块 3 种模块，在硬件设计-详细设计中将进一步介绍。



工程机器人系统硬件框架

2.2.2 硬件详细设计（自研）

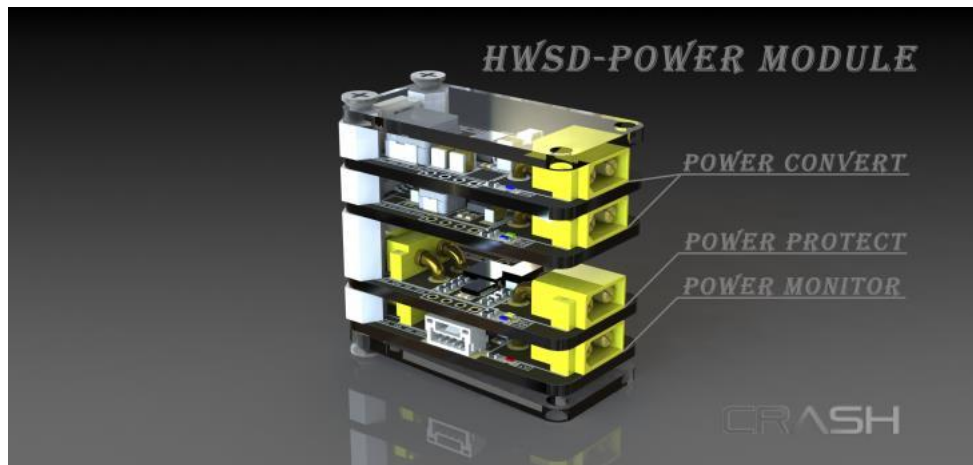


机器人硬件系统电源树

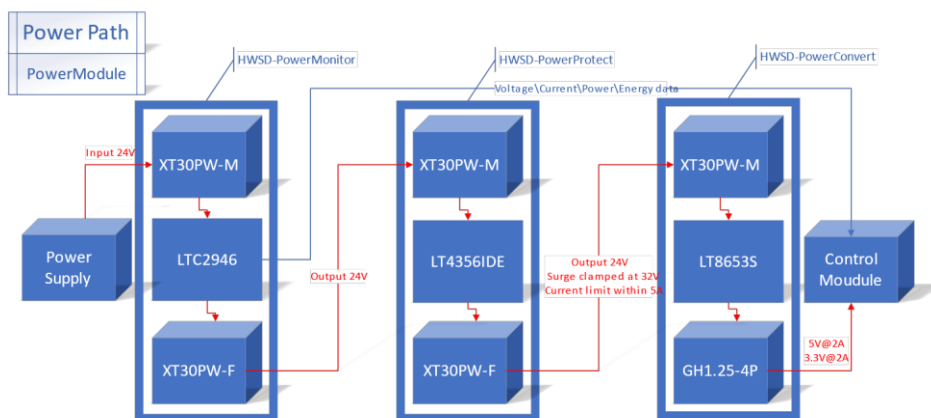
硬件系统由 24V 电池供电，接入裁判系统电源管理模块分流管理，我们自行设计的电源模块接入 MiniPC 电源输出端口，不占用底盘功率。经 LTC2946 功率监测芯片、LT4356 浪涌抑制芯片、LT8653S 双通道降压芯片，输出最大电流 2A 的 5V 和 3.3V 两个电压轨，其中 15 3.3V 电压轨给主控模块 HWSD-Core 上的 MCU STM32F405RGT6、CAN 收发器芯片 SN65HVD232，以及显示模块 HWSD-Terminal 上的 MCU STM32G030F6P6、1.14 寸 IPS 供电；5V 电压轨给遥控器接收机 DR16、USB 转串口模块 HWSD-USB2TTL 上的转换芯片 H340N、陀螺仪模块 HWSD-Gyroscope 上的 LDOLP5907 供电，LP5907 单独产生一个最大电流 250mA 的 3.3V 电压轨给陀螺仪芯片 HI229 供电。

2.2.2.1 电源模块：

电源是硬件的起点。电源优劣决定后续模块能否正常实现功能。在持续多个赛季的针对电源稳定性优化的研发过程中，我们逐渐总结设计出一套适用于机器人的电源系统，包括电源监测、电源保护、电源转换 3 类模块。本机器人系统使用的电源模块组成，电源路径见下图。

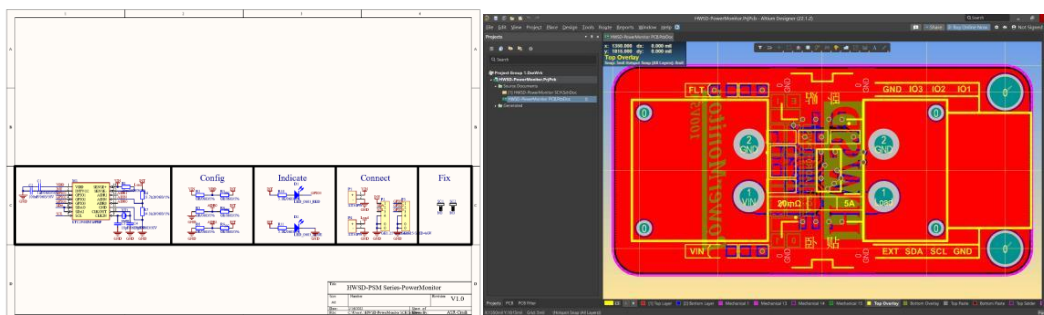


硬件标准化电源模块



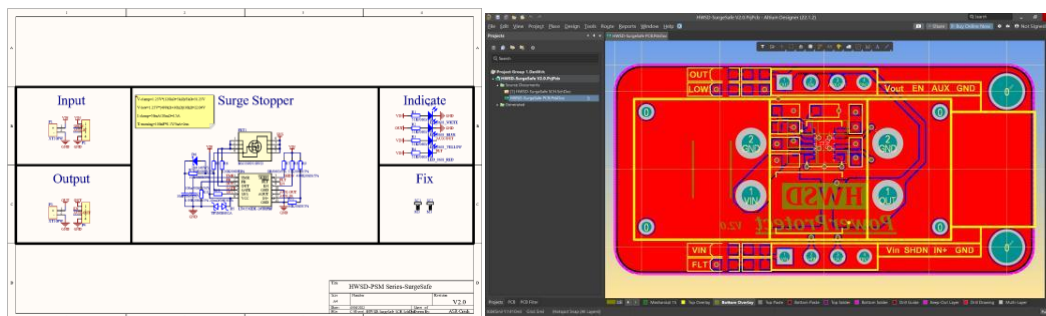
硬件标准化电源模块 Power Path

电源监测模块基于 LTC2946 电源监测芯片设计，该芯片支持 0-100V 电压输入，可实现电压、电流、功率、能耗、电荷 5 种电源数据的测量，并通过 I2C 通信传回数据。模块原理图、PCB 如下图所示。



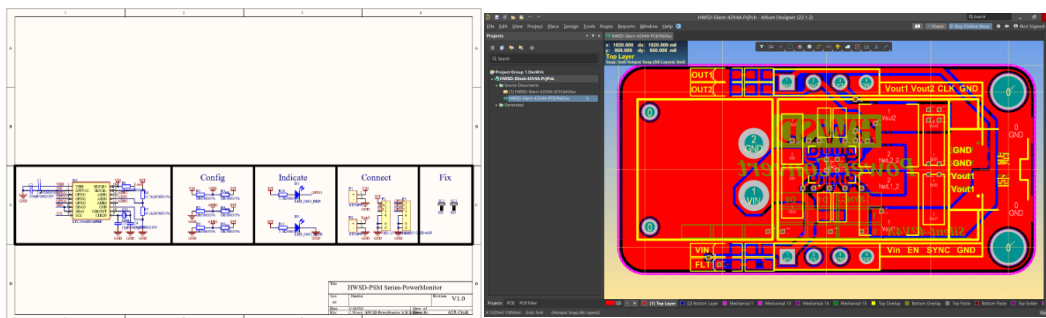
HWSD-PowerMonitor 原理图（左） PCB（右）

电源保护模块基于 LT4356 浪涌抑制芯片设计，通过控制 MOSFET BSC098N10NS5 的导通实现对高达 80V 浪涌电压的抑制，非常适合多电机的机器人系统。模块原理图、PCB 如下图所示。



HWSD-PowerProtect 原理图（左） PCB（右）

电源转换模块基于 LT4356 浪涌抑制芯片设计，通过控制 MOSFET BSC098N10NS5 的导通实现对高达 80V 浪涌电压的抑制，非常适合多电机的机器人系统。模块原理图、PCB 如下图所示。



HWSD-PowerProtect 原理图（左） PCB（右）

2.2.2.2 控制模块：

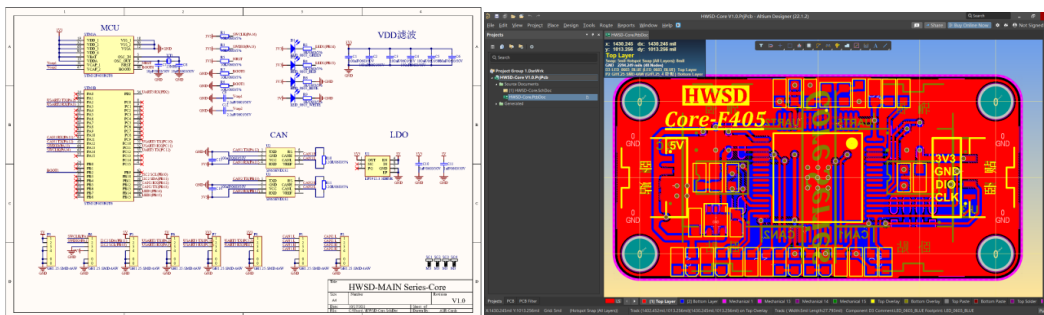
控制模块是机器人的控制核心。控制模块包括核心模块 HWSD-Core、调试模块

HWSD-DAPLink、显示模块 HWSD-IPS 3 个模块组成。控制模块渲染图如下图所示。



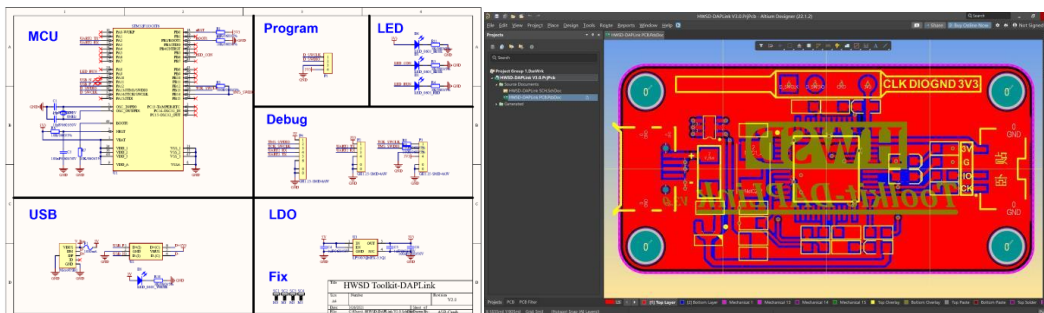
硬件标准化控制模块

核心模块 HWSD-Core 是从 HWSD-MAIN 发展而来的精简版，针对只有单一类型执行器 (CAN 电机) 的机器人系统设计，仅提供 CAN1、CAN2、UART2、UART3、UART5、I2C2、SWD 7 个功能接口，以此获得 40mm*20mm 的迷你尺寸，其原理图、PCB 如下图所示。



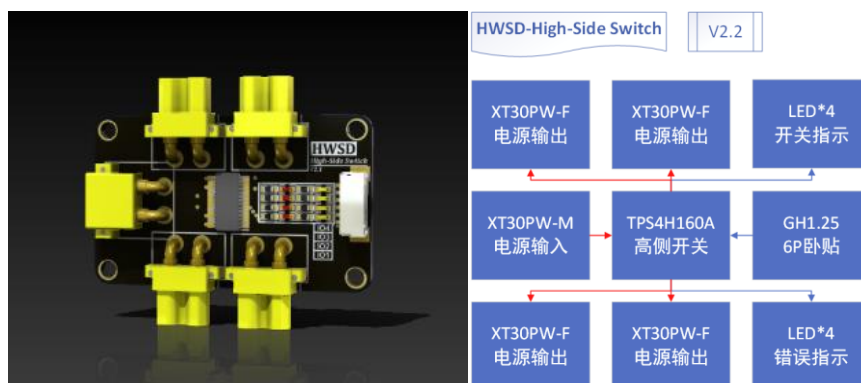
HWSD-Core 原理图 (左) PCB (右)

调试模块 HWSD-DAPLink 使用了硬件标准化定义的 GH1.25 4Pin 信号接口，替代使用排针接口的传统 ST-Link，以提高连接稳定性和调试效率。此模块原理图参考了 DAPLink 开源项目。模块原理图、PCB 如下图所示。



HWSD-DAPLink 原理图 (左) PCB (右)

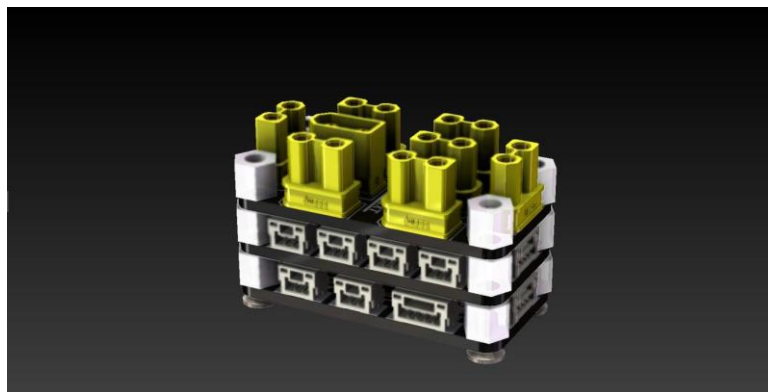
HWSD-高侧开关模块是工程机器人为了控制气泵所设置的开关，4 通道高侧开关模块，用于电磁阀等负载通电控制。其概念图、硬件框图如下所示。



HWSD-高侧开关模块概念图（左） 硬件框图（右）

2.2.2.3 电机模块：

电机模块包括 1 输入 7 输出 HWSD-XT2XT 电源扩展模块、2 输入 8 输出 HWSD-GH2GH 信号扩展模块以及作为 CAN 冗余备份的 HWSD-Core-F042 核心模块。



硬件标准化电机模块

上述电路板均采用统一的 40mm*20mm 尺寸和 M3 固定孔位，可实现机械友好型层叠安装。

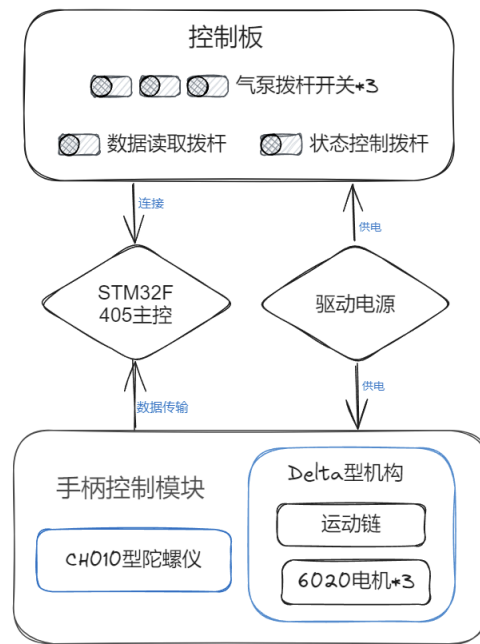
2.2.3 关键器件选型

主控模块 HWSO-Core 的 MCU 采用 STM32F405RGT6, 连续多个赛季以来我们始终使用这款芯片进行自研主控板设计与嵌入式开发, 具有良好的硬件和软件基础积淀; 另一方面, 因为较早确定了该芯片的选型方案, 我们于上个赛季采购了一批该型号芯片作为库存, 当前库存依然可以维持本赛季的研发工作, 很大程度上节省了本赛季硬件开支。

2.2.4 自定义控制器模块

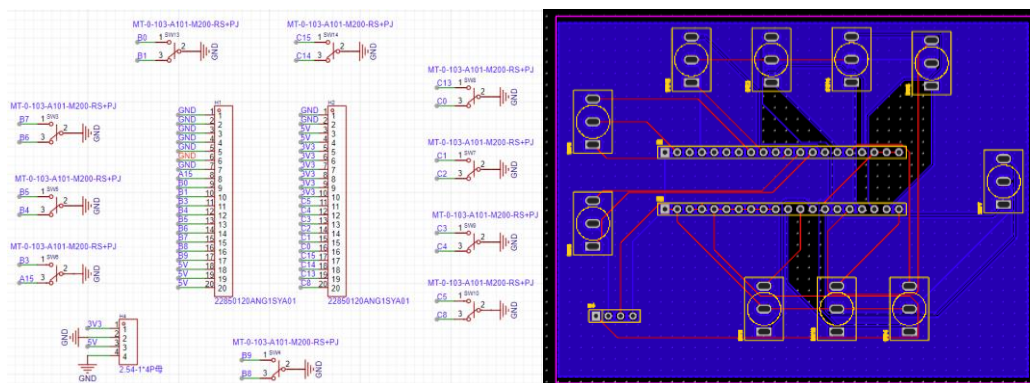
2.2.4.1 Delta 型自定义控制器

整个 Delta 型自定义控制器的设计可以分为两个模块, 一个是对六轴机械臂的位置角度控制模块, 这部分借助 delta 原理获取三个 6020 电机的关节角度值加上 CH010 型陀螺仪获取 pitch, roll, yaw 三轴角度值, 一共 6 个数据, 这 6 个数据就是控制六轴机械臂的位置角度的关键, 另一个就是按钮状态控制, 有若干个拨杆开关, 三个通过 GPIO 控制三个气泵的开关, 一个是数据读取开关, 用于读取前面控制机械的 6 个数据, 还有两个开关为状态控制, 用于输出控制器下的不同状态, 比如复位或者紧急停止。大致系统框架可见下图。



Delta 型自定义控制器系统框架

控制板模块设有多个拨杆开关，一个用于手动触发数据读取功能，将当前的机械臂位置和角度数据发送至控制器，以使用户实时监测机械臂的状态或进行后续分析处理。三个通过 GPIO 控制气泵开关，再加上两个状态切换拨杆，均为三档开关设计，其原理图、PCB 如下图所示。



控制板 原理图（左） PCB（右）

CH0X0 系列是利用 MEMS 惯性器件、地磁场、感知物体姿态信息的 IMU/VRU/AHRS 模块，搭载了自主研发的扩展卡尔曼滤波和 IMU 噪声动态分析技术，可以满足高动态下姿态角的精度。手柄控制模块集成 CH040 型陀螺仪，用于获取 pitch、roll、yaw 三轴角度值。

并联结构通过运动链连接三个 6020 电机的编码器，用于获取关节角度值。CH040 型陀螺仪如下图所示。

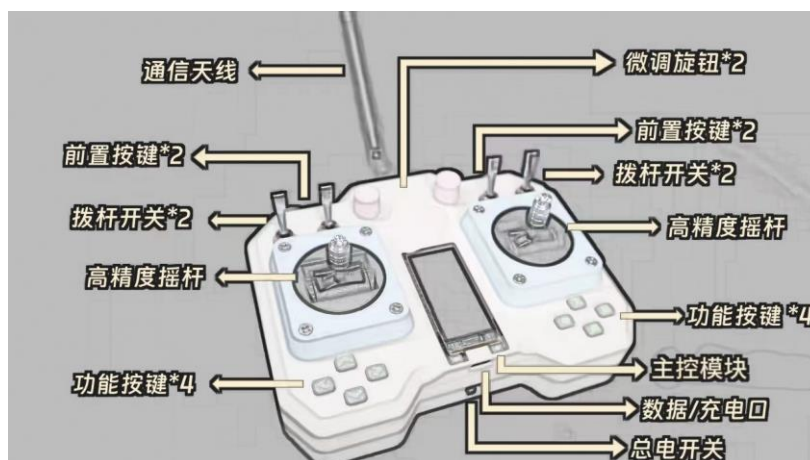


CH040 型陀螺仪

2.2.4.2 万能遥控器

功能配置

该遥控器基于 STM32F103 的高性能芯片，有丰富的配置（见下图），它设有 4 个常规按键，4 个拨杆开关，2 个电位器旋钮，2 个高精度航模摇杆，8 个功能按键，2 个板载按键，以及 1 个 CH040 电子陀螺仪。



遥控器布局图

器件选择

主控模块选取 STM32F103C8T6 最小系统板，如下图所示。STM32F103C8T6 是意法半导体（STMicroelectronics）生产的一款 32 位 ARM 微控制器，采用 Cortex-M3 内核。它集成了定时器、CAN、ADC、SPI、I2C、USB、UART 等多种外设功能。最小系统板是基于 STM32F103C8T6 芯片设计的开发板，包含了芯片的主要外设和电路，可以直接连接到计算机进行编程和调试。



STM32F103C8T6 实物图

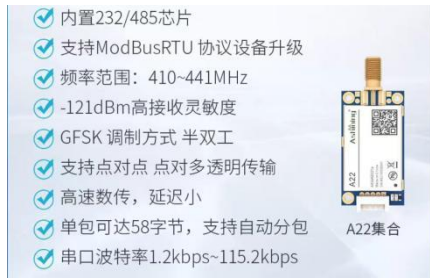
遥感是遥控器的核心部分，在这里选择航模用摇杆，精度更高，可以实现多轴控制，如下图所示。



航模遥感实物图

陀螺仪也是采用 CH040 系列。

通信模块选用 E3A-DTU-500, 它是一款 433MHz 频率的无线数传电台, 同时具有 RS232 和 RS485 接口。该通信模块常用于航模遥控中, 稳定性高, 该方式的稳定性高, 功耗很低。



无线通信模块

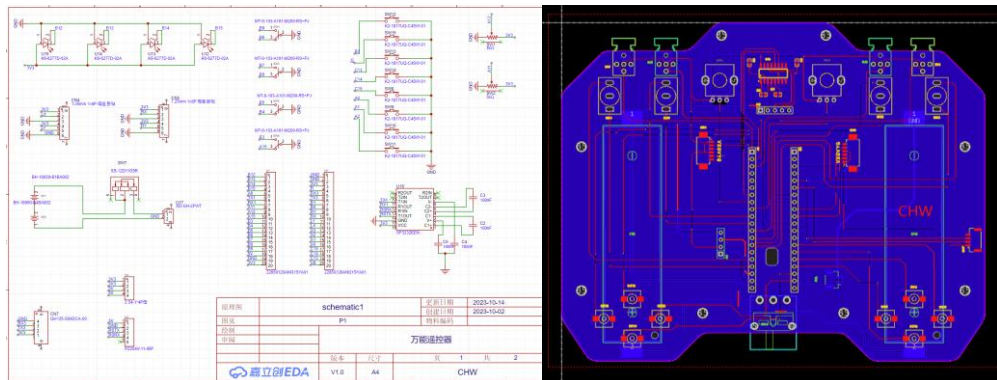
电源部分选择了 18650 锂电池，可以直接安装在电池座上，而电池座（见图 9）可以焊接到电路板上，这样就解决了电池固定的问题。



18650 锂电池座

电路设计

电路设计依据所需功能配置，使制作成品尽可能符合手感，具体如下图所示。



遥控器 原理图（左） PCB（右）

2.3 软件设计

2.3.1 系统架构

工程车作为本赛季的重要机器人之一，其代码框架仍使用厦门大学 RCS 战队自主搭建的 FreeRtos 代码框架，来进行数据调度和进程管理，这套代码框架解决了整个机器人各个传感器，控制器，执行器的工作流，上手简单，封装完备，充分利用单片机性能处理机器人设计的各种需求。本队的代码框架上层为 C++编写，大部分数据结构抽象成了类，进一步降低了开发者上手的难度，在使用一些设备或者软件资源时，直接调用封装好的类函数即可。

2.3.1.1 Main 函数

Main 函数用来在初始化阶段完成各种硬件外设的初始化以及所有数据结构的初始化和消息处理类的注册。task.Init();初始化成功之后开始执行控制程序。

```
int main(void)
{
    SystemClockConfig();
    delay.Init(168);
    //delay.delay_ms(10000);
    HAL_Init();
    can1.Init(CAN1,1);
    can2.Init(CAN2,2);
    timer.Init(BASE, TIM3, 1000).BaseInit();
    imu_pantile.Init(&uart5, UART5, 115200, CH010);
    rc.Init(&uart2, USART2, 115200);
    para.Init();
    ctrl1.Init(
    {
        &can2_motor[0], &can2_motor[1], &can2_motor[2], &can2_motor[3],&can2_motor[4],
        &can2_motor[5]
    });
    led1.Init({ GPIOB,GPIO_PIN_14 });
    led2.Init({ GPIOB,GPIO_PIN_15 });
    led3.Init({ GPIOC,GPIO_PIN_15 });
    task.Init();
    for (;;)
    ;
}
```

TASK.Init 函数用于创建开始任务，在开始任务 start_task 中会创建电机更新任务、CAN 线数据发送任务、运动学解算及信息更新任务，任务创建完成后会删除开始任务。

```

void TASK::Init()
{
    //创建开始任务
    xTaskCreate((TaskFunction_t)start_task,          //任务函数
               (const char*)"start_task",           //任务名称
               (uint16_t)START_STK_SIZE,             //任务堆栈大小
               (void*)NULL,                          //传递给任务函数的参数
               (UBaseType_t)START_TASK_PRIO,         //任务优先级
               (TaskHandle_t*)&StartTask_Handler);   //任务句柄
    vTaskStartScheduler();                          //开启任务调度
}
void start_task(void* pvParameters)
{
    taskENTER_CRITICAL();                          //进入临界区
    //创建任务

    xTaskCreate((TaskFunction_t)LedTask,
               (const char*)"LedTask",
               (uint16_t)LED_STK_SIZE,
               (void*)NULL,
               (UBaseType_t)LED_TASK_PRIO,
               (TaskHandle_t*)&LedTask_Handler);
}

```

```

    xTaskCreate((TaskFunction_t)MotorUpdateTask,
               (const char*)"MotorUpdateTask",
               (uint16_t)MOTOR_STK_SIZE,
               (void*)NULL,
               (UBaseType_t)MOTOR_TASK_PRIO,
               (TaskHandle_t*)&MotorTask_Handler);

    xTaskCreate((TaskFunction_t)CanTransimtTask,
               (const char*)"CanTransimtTask",
               (uint16_t)CANTX_STK_SIZE,
               (void*)NULL,
               (UBaseType_t)CANTX_TASK_PRIO,
               (TaskHandle_t*)&CanTxTask_Handler);

    xTaskCreate((TaskFunction_t)ControlTask,
               (const char*)"ControlTask",
               (uint16_t)CONTROL_STK_SIZE,
               (void*)NULL,
               (UBaseType_t)CONTROL_TASK_PRIO,
               (TaskHandle_t*)&ControlTask_Handler);

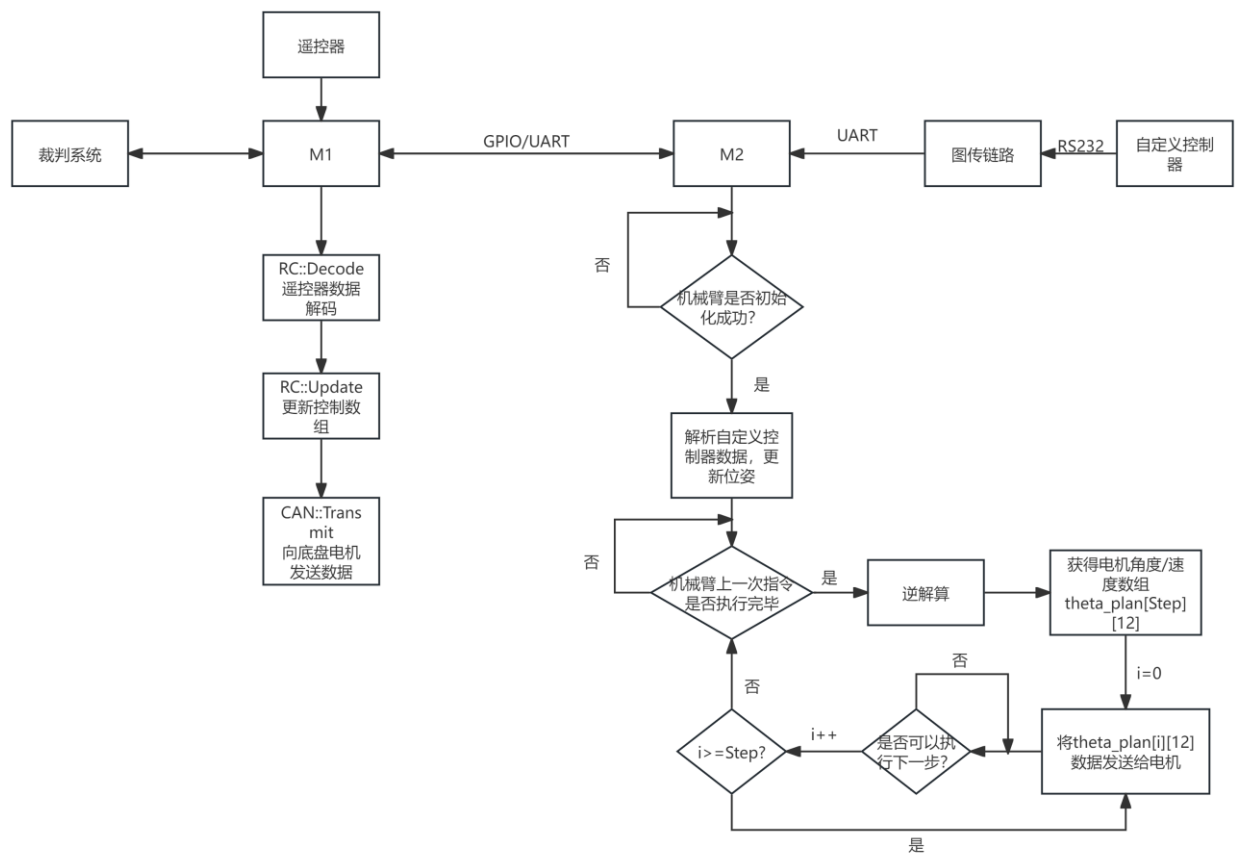
    vTaskDelete(StartTask_Handler); //删除开始任务
    taskEXIT_CRITICAL();             //退出临界区
}

```

2.3.1.2 机器人更新

为了使控制器与机械臂之间通信更加快速，机械臂控制延迟更低，我们的工程机器人采用了双主控方案，一块主板负责控制底盘以及与裁判系统进行数据交互，另一块主板只进行自定义控制器数据的解析与机械臂的控制，两块主板间通过 GPIO 及 UART 串口进行通信。

具体的更新流程拓扑图如下：



下面主要介绍机械臂的控制代码逻辑

DecodeTask 函数主要是对接收到的数据进行解码。rc.Decode 是获取自定义控制器数据的函数；

imu_pantile.Decode 用于解码车上陀螺仪的数据；htmotor[i].State_Decode 用于解码电机反馈的数据，htmotor 中存有电机的速度，角度位置，加速度等数据。

```

void DecodeTask(void* pvParameters)
{
    while (true)
    {
        rc.Decode();
        imu_pantile.Decode();
        for (int i = 0; i < 6; i++)//数据解码更新
        {
            htmotor[i].State_Decode(can1, can1.jointidata);
        }
        //theta_true[0] = cubemotor[0].pos;
        for (int i = 0; i < 6; i++) {
            if(i==1||i==2)theta_true[i] = -htmotor[i].pos;
            else
                theta_true[i] = htmotor[i].pos;
        }
        vTaskDelay(4);
    }
}

```

ControlTask 用于更新控制数组和初始化机械臂。rc.Update 函数会根据 rc.Decode 获得的数据更新标志位、机械臂期望位姿、气泵启停等相应的控制数组。如果机械臂电机未初始化，htmotor[0].HTmotorinit 函数会向所有电机发送使能信息，并执行 init_arm 电机初始化函数，解算出从当前位置到初始位置的角度/速度控制数组 theta_plan；如果电机已经初始化，则会进入 call_arm 函数，call_arm 函数会先判断机械臂是否已经到达上次期望位姿，如果是则会根据新的期望位姿调用逆解算函数 inverse_curve 获取 theta_plan。do_next_step 函数用于判断电机是否能够执行 theta_plan[i][12]中的 i+1 步，如果是则会将 theta_plan[i+1][12]中对应的角度/速度数据赋给六个关节电机的 htmotor[k].setSpeed 和 htmotor[k].setPos。


```

void ControlTask(void* pvParameters)
{
    while (true)
    {
        rc.Update();
        if (!start_flag && angle_right()) {
            htmotor[0].HTmotorinit();
            //cubemotor[0].CUBEmotorinit();
            init_arm();
            start_flag = 1;
        }
        else if (angle_right() && motor_control) {
            call_arm();
        }
        do_next_step();
        vTaskDelay(1);
    }
}

```

MotorUpdateTask 用于将 htmotor[k].setSpeed 和 htmotor[k].setPos 中的数据传入 CAN 总线发送缓冲区，准备向电机发送数据。

```

void MotorUpdateTask(void* pvParameters)
{
    while (1)
    {
        for (int i = 0; i < 6; i++)
        {
            htmotor[i].HTmotor_Ontimer(can1, Kp, Kd, can1.jointpdata[i]);
        }
        vTaskDelay(4);
    }
}

```

CanTransimtTask 通过计时器，每 1ms 向一个电机发送数据，6ms 一轮回。

```

void CanTransimtTask(void* pvParameters)
{
    while (true)
    {
        switch ((task.counter++) % 6)
        {
            case 0:
                htmotor[1].HTmotor_transmit(2);
                cubemotor[0].CUBEmotor_transmit(1);
                break;
            case 1:
                htmotor[2].HTmotor_transmit(3);
                break;
            case 2:
                htmotor[3].HTmotor_transmit(4);
                break;
            case 3:
                htmotor[4].HTmotor_transmit(5);
                break;
            case 4:
                htmotor[5].HTmotor_transmit(6);
                break;
            case 5:
                htmotor[0].HTmotor_transmit(1);
                break;
            default:
                break;
        }
        vTaskDelay(1);
    }
}

```

2.3.1.3 通讯设计

自定义控制器与机械臂主控之间的通信协议参照官方裁判系统串口协议编写，数据包格式如下：

frame_header	cmd_id	data	frame_tail
5-byte	2-byte	n-byte	2-byte, CRC16, 整包校验

裁判系统串口通讯协议

其中 cmd_id=0x302,n=30。frame_header 具体内容如下：

SOF	data_length	seq	CRC8
1-byte	2-byte	1-byte	1-byte

frame_header

其中 SOF=0xA5,data_length=30。

自定义控制器端通过定时器以 30Hz 频率调用 Control 函数发送数据，Get_Graph_buffer 函数用于添加帧头和 CRC 校验值以生成最终通过 UART 串口发送出去的 Graph_buffer。

```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
    if(htim->Instance == TIM2)
        Control();
}
void Control()
{
    .....// 将要发送的数据编码存入tx_buffer
    Get_Graph_buffer(Graph_buffer, tx_buffer);
    HAL_UART_Transmit_DMA(&huart1, Graph_buffer,sizeof(Graph_buffer));
}
```

机械臂主控端使用 DMA 从 UART 串口自动接收数据，接收到的数据将存入 m_frame, task 创建的 ControlTask 任务中的 rc.Decode 便是负责从 m_frame 中解析数据

```
void RC::Decode()
{
    pd_Rx = xQueueReceive(*queueHandler, m_frame, NULL);
    if(m_frame[0]==0xA5 && m_frame[1]==0x1E && m_frame[2]==0x00 && m_frame[3]==0x00)//校验帧头
    {
        .....//CRC校验
        .....//数据解码
    }
}
```

机械臂主控与 6 个关节电机之间通过 CAN 总线通信，main 函数中 can1.Init(CAN1,1);函数即初始化 CAN 线，需要通过 CAN 总线向某个电机发送数据时只需要调用 void HTMOTOR::HTmotor_transmit(uint32_t id)函数。

```

void HTMOTOR::HTmotor_transmit(uint32_t id)
{
    can1.Transmit(id + MOTOR_MODE, can1.jointpdata[id - 1], 8);
}
HAL_StatusTypeDef CAN::Transmit(const uint32_t ID, const uint8_t* const pData, const uint8_t len)
{
    if(can_number==1){
        hcan.pTxMsg->StdId = ID;                //设置标识符
        hcan.pTxMsg->IDE = CAN_ID_STD;           //标准帧(无扩展)
        hcan.pTxMsg->RTR = CAN_RTR_DATA;         //数据帧
        hcan.pTxMsg->DLC = len;                   //设置长度
        memcpy(hcan.pTxMsg->Data, pData, len);
        HAL_StatusTypeDef status = HAL_CAN_Transmit(&hcan, 10);
        return status;
    }
}

```

2.3.1.4 重点功能

机械臂的电控方案主要需要解决两个重要问题：

- 如何根据期望位姿获取六个关节电机的最终期望角度以及运动方案
- 如何合理的设计人机交互方案，使操作手尽可能简单的完成取矿、兑矿的流程。

对于问题 1，我们采用 M-DH 参数法通过运动学逆解算获取电机的期望角度，在获取电机的期望角度后，最重要的就是获得电机的运动方案，即获取 `theta_plan[Step][12]`。我们最终的方案如下：

首先根据电机的初始角度和期望角度获取六个电机的角度差数组 `theta[6]`，从 `theta` 中找出最大角度变量 `max_theta`，定义一个电机角度增量 `d_theta0`，

由

```

int Step = (int)(max_theta / d_theta0);
if (Step == 0) str_step = 1;

```

获取电机的总步数。

由

```
for (int i = 0; i < 6; i++)
    d_theta[i] = theta[i] / str_step;
```

获取每个电机对应的每步角度增量。

最终由

```
for (int i = 0; i < (Step); i++) {
    for (int j = 0; j < 6; j++) {
        theta_plan[i][j] = theta_init[j] + (i + 1 - Step) * d_theta[j];
    }
}
```

获取电机每一小步应该到达的角度。

在获取每个电机每小步的目标角度后，在计算每小步电机对应的速度：

首先定义一个总最大速度 speed_max,

由 theta 获取各电机的最大速度数组：

```
for (i = 0; i < 6; i++) {
    motor_speed_max[i] = speed_max * fabs(theta[i]) / PI;
    if (motor_speed_max[i] < 0.5) motor_speed_max[i] = 0.5; // 控制机械臂响应速度
}
```

根据电机 j 小步时与最终期望角度的差值利用 sin 函数获取各电机该小步时的期望速度：

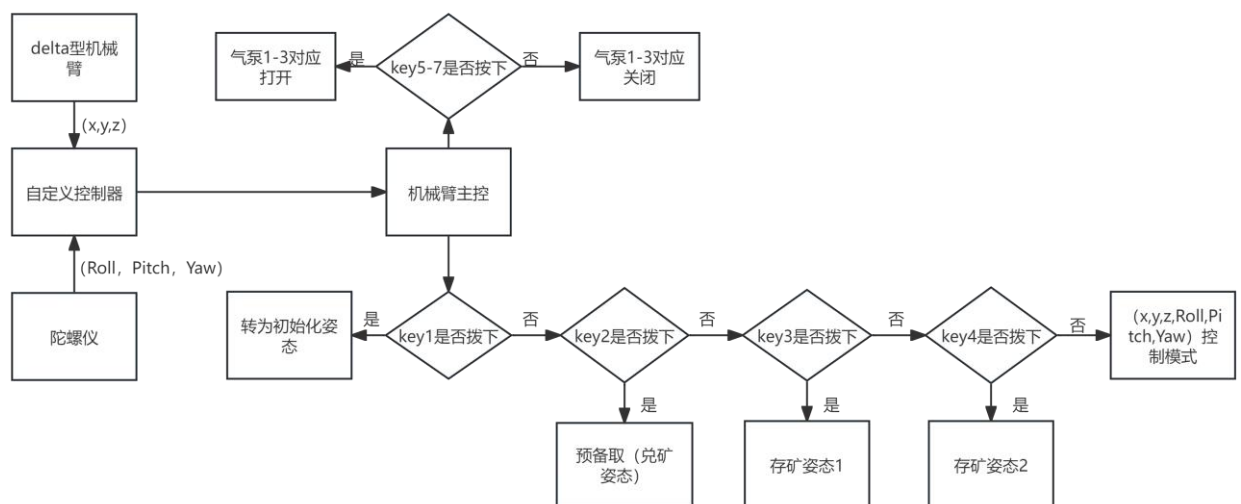
```
for (i = 0; i < Step; i++) {
    for (int j = 0; j < 6; j++) {
        if (fabs(theta[j]) < 1e-3) theta_plan[i][j + 6] = 0;
        else
            theta_plan[i][j + 6] =
                motor_speed_max[j] * (0.1 + sin((theta_plan[i][j] - theta_init_old[j])
                    * PI / (theta_plan[Step-1][j] - theta_init_old[j])));
    }
}
```

在解决第一个问题后需要考虑的就是，如何能让操作手简单的根据实际情况向机械臂输送其期望位姿，即空间坐标 (x,y,z)，空间角 (Roll, Pitch, Yaw)。一开始，我们自定义控

制器方案选用的是摇杆+陀螺仪方案，但实际测试发现，使用摇杆无法直接获取机械臂的空间坐标 (x,y,z) ，只能获取空间坐标的增量 $(\Delta x, \Delta y, \Delta z)$ ，操作手要一直使用双手操作机械臂，并不是很方便。最终，我们选择的方案是利用 delta 型机械臂+陀螺仪自定义控制器方案。Delta 型机械臂部分获取 (x,y,z) ，陀螺仪提供空间角 (Roll, Pitch, Yaw)，除此之外，自定义控制器上还设计有若干拨扭开关，以控制气泵的开关和机械臂的半自动化运动。

具体的 M-DH 参数法运动学逆解算和 delta 型机械臂的解算部分原理和推导过程将在算法设计部分具体呈现。

为使操作手取矿兑矿时操作更加简便，我们为机械臂增加了半自动模式，即提前为机械臂录入若干期望姿态(初始化姿态，存矿姿态，预备兑（取）矿姿态)。加入半自动模式后机械臂的控制逻辑如下：



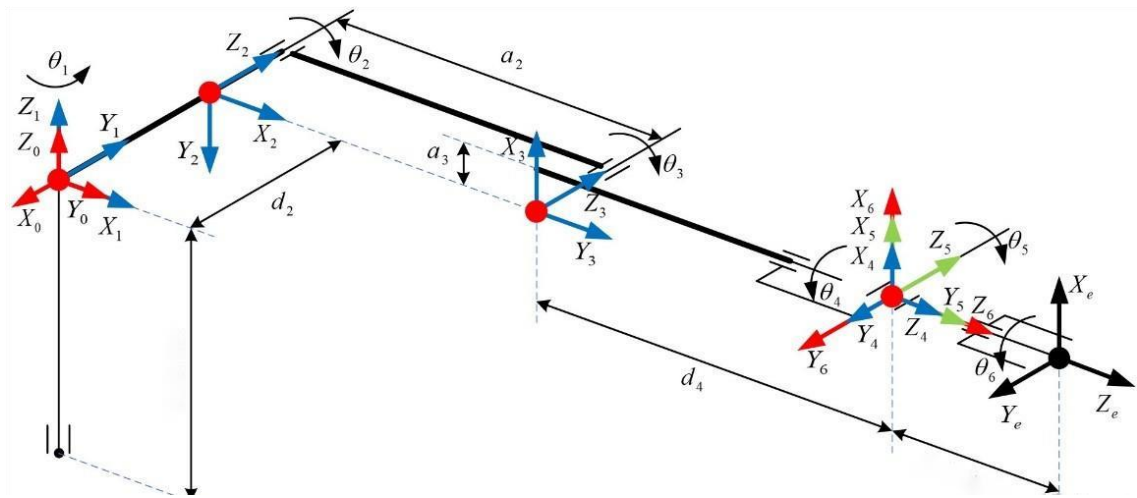
半自动模式控制逻辑图

2.4 算法设计

2.4.1 机械臂 M-DH 法逆运动学解算

2.4.1.1 M-DH 参数表

根据 M-DH 建立坐标系的方法，可以得到我们设计的类 PUMA560 架构的机械臂连杆坐标系如下：



坐标系示意图

可以得到 M-DH 参数列表如下：

i	a_{i-1}	d_i
1	0	0
2	0	d_2
3	a_2	0
4	a_3	d_3

5	0	0
6	0	0

对于我们的机械臂 $a_2=0.4318\text{m}$, $a_3=0$, $d_2=0$, $d_4=0.3035\text{m}$, 坐标系{6}与执行器末端有一段距离 $d_e=0.2865\text{m}$ 。

2.4.1.2 运动学计算

MD-H 参数法对应的齐次变换矩阵，如下式：

$${}^{i-1}_iT = Rot(z_{i-1}, \alpha_{i-1}) \cdot Trans(x_{i-1}, a_{i-1}) \cdot Rot(z_i, \theta_i) \cdot Trans(z_i, d_i) \quad (2-1)$$

$$= \begin{bmatrix} \cos \theta_i & -\sin \theta_i & 0 & a_{i-1} \\ \sin \theta_i \cos \alpha_{i-1} & \cos \theta_i \cos \alpha_{i-1} & -\sin \alpha_{i-1} & -d_i \sin \alpha_{i-1} \\ \sin \theta_i \sin \alpha_{i-1} & \cos \theta_i \sin \alpha_{i-1} & \cos \alpha_{i-1} & d_i \cos \alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

将 MD-H 参数带入式 (2-1)，得到各相邻连杆之间的齐次变换矩阵如下：

$$\begin{aligned}
{}^0_1T &= \begin{bmatrix} c\theta_1 & -s\theta_1 & 0 & 0 \\ s\theta_1 & c\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
{}^1_2T &= \begin{bmatrix} c\theta_2 & -s\theta_2 & 0 & 0 \\ 0 & 0 & 1 & d_2 \\ -s\theta_2 & -c\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
{}^2_3T &= \begin{bmatrix} c\theta_3 & -s\theta_3 & 0 & a_2 \\ s\theta_3 & c\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
{}^3_4T &= \begin{bmatrix} c\theta_4 & -s\theta_4 & 0 & a_3 \\ 0 & 0 & 1 & d_4 \\ -s\theta_4 & -c\theta_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
{}^4_5T &= \begin{bmatrix} c\theta_5 & -s\theta_5 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ s\theta_5 & c\theta_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
{}^5_6T &= \begin{bmatrix} c\theta_6 & -s\theta_6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s\theta_6 & -c\theta_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{2-2}$$

通过依次右乘齐次变化矩阵，最终可以得到机械臂末端坐标系相对于基坐标系的齐次变化矩阵 0_6T ，如下式：

$${}^0_6T = {}^0_1T \cdot {}^1_2T \cdot {}^2_3T \cdot {}^3_4T \cdot {}^4_5T \cdot {}^5_6T = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2-3}$$

式中：

$$\left\{ \begin{array}{l} n_x = c_1 [c_{23} (c_4 c_5 c_6 - s_4 s_6) - s_{23} s_5 c_6] + s_1 (s_4 c_5 c_6 + c_4 s_6) \\ n_y = s_1 [c_{23} (c_4 c_5 c_6 - s_4 s_6) - s_{23} s_5 c_6] - c_1 (s_4 c_5 c_6 + c_4 s_6) \\ n_z = -s_{23} (c_4 c_5 c_6 - s_4 s_6) - c_{23} s_5 c_6 \\ o_x = c_1 [c_{23} (-c_4 c_5 s_6 - s_4 c_6) + s_{23} s_5 s_6] + s_1 (c_4 c_6 - s_4 c_5 s_6) \\ o_y = s_1 [c_{23} (-c_4 c_5 s_6 - s_4 c_6) + s_{23} s_5 s_6] - c_1 (c_4 c_6 - s_4 c_5 s_6) \\ o_z = -s_{23} (-c_4 c_5 s_6 - s_4 c_6) + c_{23} s_5 s_6 \\ a_x = -c_1 (c_{23} c_4 s_5 + s_{23} c_5) - s_1 s_4 s_5 \\ a_y = -s_1 (c_{23} c_4 s_5 + s_{23} c_5) + c_1 s_4 s_5 \\ a_z = s_{23} c_4 s_5 - c_{23} c_5 \\ p_x = c_1 (a_2 c_2 + a_3 c_{23} - d_4 s_{23}) - d_2 s_1 \\ p_y = s_1 (a_2 c_2 + a_3 c_{23} - d_4 s_{23}) + d_2 c_1 \\ p_z = -a_3 s_{23} - a_2 s_2 - d_4 c_{23} \end{array} \right.$$

$$\left\{ \begin{array}{l} c_{23} = c_2 c_3 - s_2 s_3 \\ s_{23} = c_2 s_3 + s_2 c_3 \end{array} \right.$$

2.4.1.3 求逆运动学代数解

代数解的核心思想为：将式 (2-3) 依次左乘 ${}^{i-1}_i T^{-1}$ 使得等式右边不存在参数从而将方程解耦，具体步骤如下：

第一步：求 θ_1

将式 (2-3) 左乘 ${}^0_1 T^{-1}$ ，则有以下式：

$$\begin{aligned} & {}^0_1 T^{-1}(\theta_1) \cdot {}^0_6 T = {}^1_2 T(\theta_2) \cdot {}^2_3 T(\theta_3) \cdot {}^3_4 T(\theta_4) \cdot {}^4_5 T(\theta_5) \cdot {}^5_6 T(\theta_6) \\ & \begin{bmatrix} c_1 & s_1 & 0 & 0 \\ -s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = {}^1_6 T(Left) \\ & {}^1_6 T(Right) = \begin{bmatrix} {}^1n_x & {}^1o_x & {}^1a_x & {}^1p_x \\ {}^1n_y & {}^1o_y & {}^1a_y & {}^1p_y \\ {}^1n_z & {}^1o_z & {}^1a_z & {}^1p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \quad (2-4)$$

式中：

$$\left\{ \begin{array}{l} {}^1n_x = c_{23} (c_4 c_5 c_6 - s_4 s_6) - s_{23} s_5 c_6 \\ {}^1n_y = -s_4 c_5 c_6 - c_4 s_6 \\ {}^1n_z = -s_{23} (c_4 c_5 c_6 - s_4 s_6) - c_{23} s_5 c_6 \\ {}^1o_x = -c_{23} (c_4 c_5 s_6 + s_4 c_6) + s_{23} s_5 s_6 \\ {}^1o_y = s_4 c_5 s_6 - c_4 c_6 \\ {}^1o_z = s_{23} (c_4 c_5 s_6 + s_4 c_6) + c_{23} s_5 s_6 \\ {}^1a_x = -c_{23} c_4 s_5 - s_{23} c_5 \\ {}^1a_y = s_4 s_5 \\ {}^1a_z = s_{23} c_4 s_5 - c_{23} c_5 \\ {}^1p_x = a_2 c_2 + a_3 c_{23} - d_4 s_{23} \\ {}^1p_y = d_2 \\ {}^1p_z = -a_2 s_{23} - a_2 s_2 - d_4 c_{23} \end{array} \right.$$

式 (2-4) 左边只含有参数所以需要从等式右边找到一个只含有或者为常数的元素，可以找到该元素位于第二行第四列。令等式左边 $L(2, 4)$ 与右边 $R(2,4)$ 的对应元素相等，利用三角代换，则有：

$$\left\{ \begin{array}{l} L(2, 4) = R(2, 4) \\ L(2, 4) = -s\theta_1 \cdot p_x + c\theta_1 \cdot p_y \\ R(2, 4) = d_2 \end{array} \right. \quad (2-5)$$

式中： $p_x = \rho \cos \phi$, $p_y = \rho \sin \phi$ ；且有 $\rho = \sqrt{p_x^2 + p_y^2}$, $\phi = \text{atan2}(p_y, p_x)$ 。

于是有：

$$\left\{ \begin{array}{l} \sin(\phi - \theta_1) = d_2 / \rho \\ \cos(\phi - \theta_1) = \pm \sqrt{1 - (d_2 / \rho)^2} \\ \phi - \theta_1 = \text{atan2} \left[d_2 / \rho, \pm \sqrt{1 - (d_2 / \rho)^2} \right] \end{array} \right. \quad (2-6)$$

可以求得 θ_1 的表达式如下：

$$\theta_1 = \text{atan2}(p_y, p_x) - \text{atan2}\left(d_2, \pm \sqrt{p_x^2 + p_y^2 - d_2^2}\right) \quad (2-7)$$

式中：正负号对应 θ_1 的两个可能的解。

第二步：求 θ_3

当选定 θ_1 的一个解之后，继续求解 θ_2 或 θ_3 。令式 (2-4) 两端的元素 L(1, 4) 和 R(3, 4) 分别对应相等，并结合式 (2-5)，则有：

$$\begin{cases} c_1 p_x + s_1 p_y = a_2 c_2 + a_3 c_{23} - d_4 s_{23} \\ p_x = -a_2 s_{23} - a_2 s_2 - d_4 c_{23} \\ d_2 = -s\theta_1 p_x + c\theta_1 p_y \end{cases} \quad (2-8)$$

将上式中的三个等式的左右两边分别进行平方后，再相加求和，得到：

$$a_3 c_3 - d_4 s_3 = k \quad (2-9)$$

式中： $\frac{p_x^2 + p_y^2 + p_z^2 - a_2^2 - a_3^2 - d_2^2 - d_4^2}{2a_2}$

通过三角代换，求得：

$$\theta_3 = \text{atan2}(a_3, d_4) - \text{atan2}\left(k, \pm \sqrt{a_3^2 + d_4^2 - k^2}\right) \quad (2-10)$$

式中：正、负号对应 θ_3 的两种可能的解。

第三步：求解 θ_2

已经求出 θ_1 和 θ_3 之后，可以通过将式 (2-4) 左乘一个包含 θ_1 、 θ_2 和 θ_3 的逆变换

${}^3_0T^{-1}(\theta_1, \theta_2, \theta_3)$ ，进而求解出 θ_2 。

$$\begin{aligned}
& {}^0_3T^{-1}(\theta_1, \theta_2, \theta_3) \cdot {}^0_6T = {}^3_4T(\theta_4) \cdot {}^4_5T(\theta_5) \cdot {}^5_6T(\theta_6) \\
& \begin{bmatrix} c_1 c_{23} & s_1 c_{23} & -s_{23} & -a_2 c_3 \\ -c_1 s_{23} & -s_1 s_{23} & -c_{23} & a_2 s_3 \\ -s_1 & c_1 & 0 & -d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & u_x & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = {}^3_6T(Left) \\
& {}^3_6T(Right) = \begin{bmatrix} c_4 c_5 c_6 - s_4 s_6 & -c_4 c_5 c_6 - s_4 c_6 & -c_4 s_5 & a_3 \\ s_5 s_5 & -s_5 s_6 & c_5 & d_4 \\ -s_4 s_5 s_6 - c_4 s_6 & s_4 c_5 s_6 - c_4 c_6 & s_4 s_5 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{2-11}$$

从式 (2-11) 右边的矩阵可以看出只有 $R(1, 4)$ 和 $R(2, 4)$ 为常数, 所以令左右两边(1, 4)和(2, 4)处的元素分别对应相等, 得到下式:

$$\begin{cases} c_1 c_{23} p_x + s_1 c_{23} p_y - s_{23} p_z - a_2 c_3 = a_3 \\ -c_1 s_{23} p_x - s_1 s_{23} p_y - c_{23} p_z + a_2 s_3 = d_4 \end{cases} \tag{2-12}$$

联立求解得:

$$\begin{cases} s_{23} = \frac{(-a_3 - a_2 c_3) p_z + (c_1 p_x + s_1 p_y)(a_2 s_3 - d_4)}{p_z^2 + (c_1 p_x + s_1 p_y)^2} \\ c_{23} = \frac{(-d_4 + a_2 s_3) p_z + (c_1 p_x + s_1 p_y)(a_2 c_3 + a_3)}{p_z^2 + (c_1 p_x + s_1 p_y)^2} \end{cases} \tag{2-13}$$

s_{23} 和 c_{23} 表达式的分母相同且为正, 于是有:

$$\theta_{23} = \theta_2 + \theta_3 = \text{atan2} \left[\frac{-(a_3 + a_2 c_3) p_z + (c_1 p_x + s_1 p_y)(a_2 s_3 - d_4)}{(-d_4 + a_2 s_3) p_z + (c_1 p_x + s_1 p_y)(a_2 c_3 + a_3)}, \right] \tag{2-14}$$

根据 θ_1 和 θ_3 解的四种可能组合, 由式 (2-14) 可以得到对应的四种可能值 θ_{23} , 于是可以得到 θ_2 的四种可能的解:

$$\theta_2 = \theta_{23} - \theta_3 \tag{2-15}$$

式中 θ_2 取与 θ_3 相对应的值。

第四步: 求 θ_4

此时, 式 (2-11) 左边均为已知, 通过观察矩阵元素, 可以通过(1, 3)和(3, 3)处两个元素的比值求出 θ_4 , 所以令等式两边的元素(1, 3)和(3, 3)分别对应相等, 则有:

$$\begin{cases} a_x c_1 c_{23} + a_y s_1 c_{23} - a_z s_{23} = -c_4 s_5 \\ -a_x s_1 + a_y c_1 = s_4 s_5 \end{cases} \quad (2-16)$$

分两种情况：

当 $s_5 \neq 0$ 时，可以求出 θ_4 的表达式：

$$\theta_4 = \text{atan2}(-a_x s_1 + a_y c_1, -a_x c_1 c_{23} - a_y s_1 c_{23} + a_z s_{23}) \quad (2-17)$$

当 $s_5 = 0$ 时，机械臂处于奇异位形。此时，关节轴 4 和 6 重合，只能求出 θ 与 θ 的和或差。

处于奇异位形时，可以任意选取 θ_4 的值，再计算对应的 θ_6 。

第五步：求 θ_5

根据求出的 θ_4 ，可以进一步求出 θ_5 ，将方程 (2-3) 两边左乘一个逆变换 ${}^4T^{-1}(\theta_1, \theta_2, \theta_3, \theta_4)$ ，

则有：

$$\begin{aligned} & {}^0T^{-1}(\theta_1, \theta_2, \theta_3, \theta_4) \cdot {}^0T = {}^4T(\theta_5) \cdot {}^5T(\theta_6) \\ & \begin{bmatrix} c_1 c_{23} c_4 + s_1 s_4 & s_1 c_{23} c_4 - c_1 s_4 & -s_{23} c_4 & -a_2 c_3 c_4 + d_2 s_4 - a_3 c_4 \\ -c_1 c_{23} s_4 + s_1 c_4 & -s_1 c_{23} s_4 - c_1 c_4 & s_{23} s_4 & a_2 c_3 s_4 + d_2 c_4 + a_3 s_4 \\ -c_1 s_{23} & -s_1 s_{23} & -c_{23} & a_2 s_3 - d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = {}^4T(Left) \\ & {}^4T(Right) = \begin{bmatrix} c_5 c_6 & -c_5 s_6 & -s_5 & 0 \\ s_6 & c_6 & 0 & 0 \\ s_5 s_6 & -s_5 c_6 & c_5 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \quad (2-18)$$

式 (2-18) 左边的 $\theta_1, \theta_2, \theta_3$ 和 θ_4 都已经求出，可以观察到矩阵中仅仅包含 θ_5 的元素为

$L(1, 3)$ 和 $L(3, 3)$ ，令矩阵两边 (1, 3) 和 (3, 3) 处的元素分别对应相等，则有：

$$\begin{cases} a_x (c_1 c_{23} c_4 + s_1 s_4) + a_y (s_1 c_{23} c_4 - c_1 s_4) + a_z (-s_{23} c_4) = -s_5 \\ a_x (-c_1 s_{23}) + a_y (-s_1 s_{23}) + a_z (-c_{23}) = c_5 \end{cases} \quad (2-19)$$

于是得到 θ_5 的表达式：

$$\theta_5 = \text{atan2}(s_5, c_5) \quad (2-20)$$

第六步：求 θ_6 ：

将方程 (2-3) 两边左乘逆变换 ${}^5_0T^{-1}(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5)$, 则有:

$${}^0_5T^{-1}(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5) \cdot {}^0_6T = {}^5_6T(\theta_6) \quad (2-21)$$

令式 (2-21) 中(1, 1)和(3, 1)处的等式两边的元素对应相等, 可得:

$$\begin{cases} -n_x (c_1 c_{23} s_4 - s_1 c_4) - n_y (s_1 c_{23} s_4 + c_1 c_4) + n_z (s_{23} s_4) = s_6 \\ n_x [(c_1 c_{23} c_4 + s_1 s_4) c_5 - c_1 s_{23} s_5] + n_y [(s_1 c_{23} c_4 - c_1 s_4) c_5 - s_1 s_{23} s_5] \\ -n_z (s_{23} c_4 c_5 + c_{23} s_5) = c_6 \end{cases} \quad (2-22)$$

从而求出 θ_6 的表达式:

$$\theta_6 = \text{atan2}(s_6, c_6) \quad (2-23)$$

至此, 六个关节电机的角度已全部求得, 不难发现, 机械臂末端的一组特定位姿, 对应的逆解有 8 组, 在得到这八组解后, 我们便需要根据机械臂的限位以及初始状态选择最优解。

我们已经获得了由 6_0T 求得角度的算法, 那么 6_0T 该怎么由(x,y,z,Roll,Pitch,Yaw)获得呢?

实际上, 当不考虑 de 时, 6_0T 表达式如下:

$${}^6_0T = \begin{bmatrix} \cos(R)*\cos(P), & \cos(R)*\sin(Y)*\sin(P) - \cos(Y)*\sin(R), & \sin(Y)*\sin(R) + \cos(Y)*\cos(R)*\sin(P), & x; \\ \cos(P)*\sin(R), & \cos(Y)*\cos(R) + \sin(R)*\sin(P)*\sin(Y), & \cos(Y)*\sin(P)*\sin(R) - \cos(R)*\sin(Y), & y; \\ -\sin(P), & \cos(P)*\sin(Y), & \cos(Y)*\cos(P), & z; \\ \theta, & \theta, & \theta, & 1]; \end{bmatrix}$$

当考虑 de 时, 利用简单的矩阵运算可以得到:

$${}^6_0T = \begin{bmatrix} \cos(R)*\cos(P), & \cos(R)*\sin(Y)*\sin(P) - \cos(Y)*\sin(R), & \sin(Y)*\sin(R) + \cos(Y)*\cos(R)*\sin(P), & -de*(\sin(Y)*\sin(R) + \cos(Y)*\cos(R)*\sin(P))+x; \\ \cos(P)*\sin(R), & \cos(Y)*\cos(R) + \sin(R)*\sin(P)*\sin(Y), & \cos(Y)*\sin(P)*\sin(R) - \cos(R)*\sin(Y), & -de*(\cos(Y)*\sin(P)*\sin(R) - \cos(R)*\sin(Y))+y; \\ -\sin(P), & \cos(P)*\sin(Y), & \cos(Y)*\cos(P), & -de*(\cos(Y)*\cos(P))+z; \\ \theta, & \theta, & \theta, & 1]; \end{bmatrix}$$

2.4.2 Matlab 仿真

为测试我们的机械臂是否能达到兑矿取矿的要求, 在实际操作前, 我们需要对机械臂位

姿的可达性测试，借助 matlab 的机器人工具箱，我们进行了以下仿真运算。

2.4.2.1 部分代码展示

```
%定义位姿变换矩阵匿名函数
T=@(theta,d,a,alpha)[cos(theta),-sin(theta), 0,a;
sin(theta)*cos(alpha), cos(theta)*cos(alpha), -sin(alpha),-d*sin(alpha);
sin(theta)*sin(alpha), cos(theta)*sin(alpha), cos(alpha), d*cos(alpha);
0, 0, 0, 1];

%解析法计算逆运动学
a2=a(3); a3=a(4);
d2=d(2); d4=d(4);
if flag
flag=0;
T_actuator = [cos(R)*cos(P), cos(R)*sin(Y)*sin(P) - cos(Y)*sin(R), sin(Y)*sin(R) + c
cos(P)*sin(R), cos(Y)*cos(R) + sin(R)*sin(P)*sin(Y), cos(Y)*sin(P)*sin(R) - cos(R)*s
-sin(P), cos(P)*sin(Y), cos(Y)*cos(P), T_actuator(3,4);
0, 0, 0, 1];
end
n_x=T_actuator(1); n_y=T_actuator(2); n_z=T_actuator(3);
o_x=T_actuator(5); o_y=T_actuator(6); o_z=T_actuator(7);
a_x=T_actuator(9); a_y=T_actuator(10); a_z=T_actuator(11);
p_x=T_actuator(13); p_y=T_actuator(14); p_z=T_actuator(15);
Theta=zeros(8,6);

m=0;
for i=1:2
for j=1:2
for k=1:2
m=m+1;
%求解theta1(为弧度)
sqr1=[sqrt(p_x^2+p_y^2-d2^2),-sqrt(p_x^2+p_y^2-d2^2)];
if(imag(sqr1(i))~=0)
Theta(m,:)=theta_init;
break;
end
ta1=atan2(p_y,p_x)-atan2(d2,sqr1(i));
%求解theta3 (弧度表示)
k1=(p_x^2+p_y^2+p_z^2-a2^2-a3^2-d2^2-d4^2)/(2*a2);
sqr3=[sqrt(a3^2+d4^2-k1^2),-sqrt(a3^2+d4^2-k1^2) ];
if(imag(sqr3(j))~=0)
Theta(m,:)=theta_init;
break;
end
```



```

ta3=atan2(a3,d4)-atan2(k1,sqr3(j));
fs23=-((a3+a2*cos(ta3))*p_z)+(cos(ta1)*p_x+sin(ta1)*p_y)*(a2*sin(ta3)-d4);
sc23=(-d4+a2*sin(ta3))*p_z+(cos(ta1)*p_x+sin(ta1)*p_y)*(a2*cos(ta3)+a3);
ta23=atan2(fs23,sc23);
%求解theta2 (弧度表示)
ta2=ta23-ta3;
%求解theta4 (弧度表示)
fs4=[ -a_x*sin(ta1)+a_y*cos(ta1),a_x*sin(ta1)-a_y*cos(ta1)];
sc4=[ -a_x*cos(ta1)*cos(ta23)-a_y*sin(ta1)*cos(ta23)+a_z*sin(ta23),a_x*cos(ta1)*cos(
ta4=atan2(fs4(k),sc4(k));
%求解theta5 (弧度表示)
fs5=-a_x*(cos(ta1)*cos(ta23)*cos(ta4)+sin(ta1)*sin(ta4))...
-a_y*(sin(ta1)*cos(ta23)*cos(ta4)-cos(ta1)*sin(ta4))...
+a_z*(sin(ta23)*cos(ta4));
sc5=a_x*(-cos(ta1)*sin(ta23))+a_y*(-sin(ta1)*sin(ta23))+a_z*(-cos(ta23));
ta5=atan2(fs5,sc5);
%求解theta6 (弧度表示)
fs6=-n_x*(cos(ta1)*cos(ta23)*sin(ta4)-sin(ta1)*cos(ta4))...
-n_y*(sin(ta1)*cos(ta23)*sin(ta4)+cos(ta1)*cos(ta4))...
+n_z*(sin(ta23)*sin(ta4));
sc6= n_x*(cos(ta1)*cos(ta23)*cos(ta4)+sin(ta1)*sin(ta4))*cos(ta5)...
-n_x*cos(ta1)*sin(ta23)*sin(ta5)...
+n_y*(sin(ta1)*cos(ta23)*cos(ta4)+cos(ta1)*sin(ta4))*cos(ta5)...
-n_y*sin(ta1)*sin(ta23)*sin(ta5)...
-n_z*(sin(ta23)*cos(ta4)*cos(ta5)+cos(ta23)*sin(ta5));
ta6=atan2(fs6,sc6);
%save

```

```

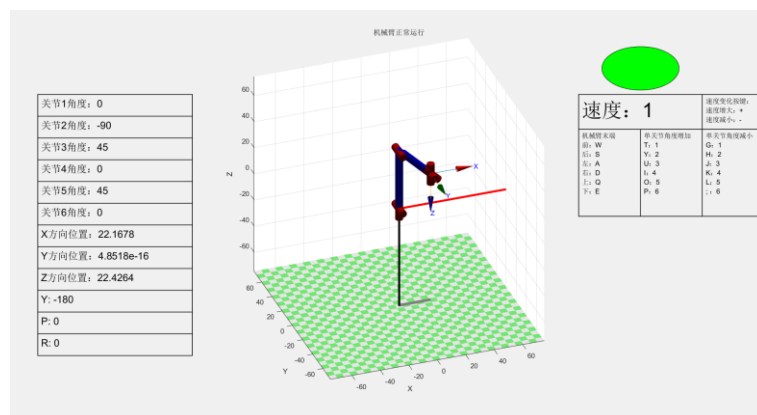
Theta(m,:)= [ta1 ta2 ta3 ta4 ta5 ta6];
end
end
End

%选择最优解
disp(Theta(:, :));
% 计算Theta中每组解与theta_init的欧几里得距离
dist = sqrt(sum((Theta - theta_init).^2, 2));
% 找出距离最小的那组解的索引
[~, index] = min(dist);

```

2.4.2.2 运行展示

运行界面：

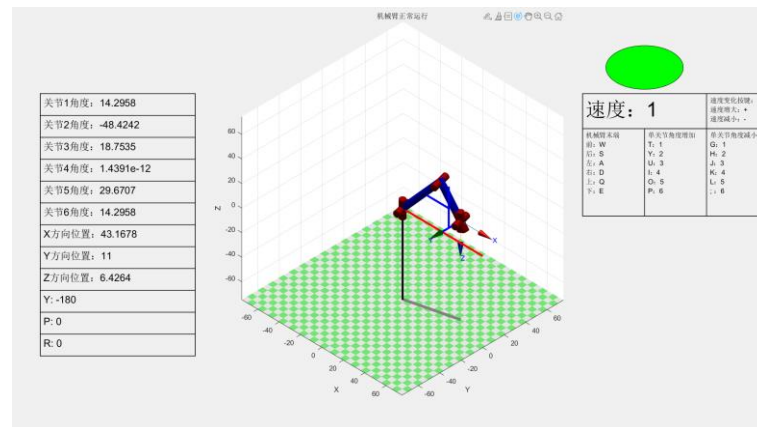


运行界面图

平移运动：

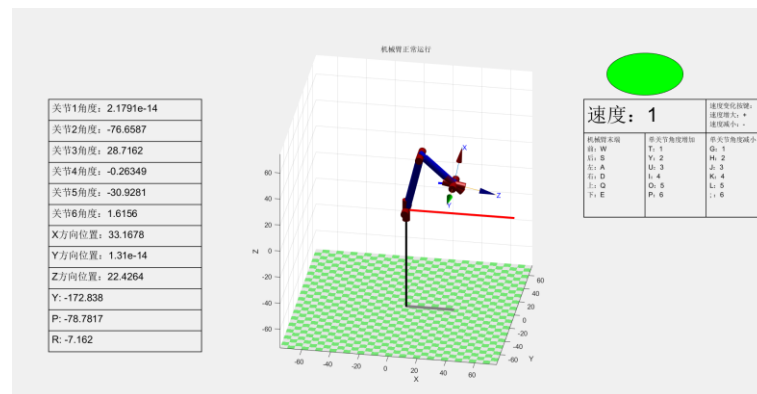
0.2055	-0.8508	0.3433	0.0000	0.5076	0.2055
0.2055	-0.8508	0.3433	-3.1416	-0.5076	-2.9360
0.2055	-5.7218	2.8642	0.0000	2.8576	0.2055
0.2055	-5.7218	2.8642	-3.1416	-2.8576	-2.9360
-2.9360	2.5802	0.3433	-3.1416	2.9235	0.2055
-2.9360	2.5802	0.3433	0.0000	-2.9235	-2.9360
-2.9360	-2.2907	2.8642	-3.1416	0.5734	0.2055
-2.9360	-2.2907	2.8642	0.0000	-0.5734	-2.9360
0.2276	-0.8481	0.3357	0.0000	0.5125	0.2276
0.2276	-0.8481	0.3357	-3.1416	-0.5125	-2.9140
0.2276	-5.7231	2.8718	0.0000	2.8513	0.2276
0.2276	-5.7231	2.8718	-3.1416	-2.8513	-2.9140
-2.9140	2.5815	0.3357	-3.1416	2.9172	0.2276
-2.9140	2.5815	0.3357	0.0000	-2.9172	-2.9140
-2.9140	-2.2934	2.8718	-3.1416	0.5783	0.2276
-2.9140	-2.2934	2.8718	0.0000	-0.5783	-2.9140
0.2495	-0.8452	0.3273	0.0000	0.5179	0.2495
0.2495	-0.8452	0.3273	-3.1416	-0.5179	-2.8921
0.2495	-5.7246	2.8802	0.0000	2.8444	0.2495
0.2495	-5.7246	2.8802	-3.1416	-2.8444	-2.8921
-2.8921	2.5830	0.3273	-3.1416	2.9103	0.2495
-2.8921	2.5830	0.3273	0.0000	-2.9103	-2.8921
-2.8921	-2.2964	2.8802	-3.1416	0.5837	0.2495
-2.8921	-2.2964	2.8802	0.0000	-0.5837	-2.8921

平移数据



当键盘按下 WSADQE 中的按键后，程序会更新机械臂的期望位姿，每次更新后都会计算机械臂六个关节电机的期望角度，每次输出 8 组解，程序会选择最优解执行。空间坐标上会留下机械臂的运行轨迹。从仿真结果来看，我们的逆解算代码基本无误。

改变空间姿态：



当键盘按下 ZXC 时，机械臂末端分别绕 Yaw、Pitch、Row 旋转，解算结果也正确。

2.4.2.3 机械臂主控端代码实现

```
void inverse0()
{
    double x = p_end[0], y = p_end[1], z = p_end[2], X = p_end[3], Y = p_end[4],
    Z = p_end[5];
    //定义一个4x4的二维数组
    double T06[4][4];

    //用原来的T06矩阵的元素赋值给数组
    T06[0][0] = cos(Z) * cos(Y);
    T06[0][1] = cos(Z) * sin(X) * sin(Y) - cos(X) * sin(Z);
    T06[0][2] = sin(X) * sin(Z) + cos(X) * cos(Z) * sin(Y);
    T06[0][3] = x;
    T06[1][0] = cos(Y) * sin(Z);
    T06[1][1] = cos(X) * cos(Z) + sin(Z) * sin(Y) * sin(X);
    T06[1][2] = cos(X) * sin(Y) * sin(Z) - cos(Z) * sin(X);
    T06[1][3] = y;
    T06[2][0] = -sin(Y);
    T06[2][1] = cos(Y) * sin(X);
    T06[2][2] = cos(X) * cos(Y);
    T06[2][3] = z;
    T06[3][0] = 0;
    T06[3][1] = 0;
    T06[3][2] = 0;
    T06[3][3] = 1;

    /*//修正吸盘至z6坐标系原点直线距离de;
    T06[0][0] = cos(Z) * cos(Y);
    T06[0][1] = cos(Z) * sin(X) * sin(Y) - cos(X) * sin(Z);
    T06[0][2] = sin(X) * sin(Z) + cos(X) * cos(Z) * sin(Y);
    T06[0][3] = -de*(sin(X) * sin(Z) + cos(X) * cos(Z) * sin(Y))+x;
    T06[1][0] = cos(Y) * sin(Z);
    T06[1][1] = cos(X) * cos(Z) + sin(Z) * sin(Y) * sin(X);
    T06[1][2] = cos(X) * sin(Y) * sin(Z) - cos(Z) * sin(X);
    T06[1][3] = -de*(cos(X) * sin(Y) * sin(Z) - cos(Z) * sin(X))+y;
    T06[2][0] = -sin(Y);
    T06[2][1] = cos(Y) * sin(X);
    T06[2][2] = cos(X) * cos(Y);
    T06[2][3] = -de*(cos(X) * cos(Y))+z;
    T06[3][0] = 0;
    T06[3][1] = 0;
    T06[3][2] = 0;
    T06[3][3] = 1;
    */
}
```

```

double n_x, n_y, n_z, o_x, o_y, o_z, a_x, a_y, a_z, p_x, p_y, p_z;
n_x = T06[0][0]; n_y = T06[1][0]; n_z = T06[2][0];
o_x = T06[0][1]; o_y = T06[1][1]; o_z = T06[2][1];
a_x = T06[0][2]; a_y = T06[1][2]; a_z = T06[2][2];
p_x = T06[0][3]; p_y = T06[1][3]; p_z = T06[2][3];
double Theta[8][6] = { 10 * PI };

int m = -1, total = 8;
int i;
int j;
int k;
for (i = 0; i < 2; i++) {
    for (j = 0; j < 2; j++) {
        for (k = 0; k < 2; k++) {
            // 求解theta1(为弧度)
            double sqr1[2] = { sqrt(p_x * p_x + p_y * p_y - d2 * d2),
                               -sqrt(p_x * p_x + p_y * p_y - d2 * d2) };
            // 如果sqr1(i)是复数
            // 计算ta1的值
            double ta1 = atan2(p_y, p_x) - atan2(d2, sqr1[i]);
            if (1(ta1 <= (PI + PIe) / 2 && ta1 >= -(PI + PIe) / 2)) {
                total = total - 1;
                continue;
            }
        }
    }
}

```

```

// 求解theta3 (弧度表示)
double k1 =
(p_x * p_x + p_y * p_y + p_z * p_z - a2 *
a2 - a3 * a3 - d2 * d2 - d4 * d4) / (2 * a2);
double sqr3[2] =
{ sqrt(a3 * a3 + d4 * d4 - k1 * k1), -sqrt(a3 *
a3 + d4 * d4 - k1 * k1) };
// 如果sqr3(j)是复数
// 计算ta3的值
double ta3 = atan2(a3, d4) - atan2(k1, sqr3[j]);
// 如果ta3大于等于pi/2或小于等于-3*pi/2
if (1(ta3 <= (PI + PIe) / 4 && ta3 >= -(PI + PIe) / 2)) {
    total = total - 1;
    continue;
}
double fs23 = -((a3 + a2 * cos(ta3)) * p_z) + (cos(ta1) *
p_x + sin(ta1) * p_y) * (a2 * sin(ta3) - d4);
double sc23 = (-d4 + a2 * sin(ta3)) * p_z + (cos(ta1) *
p_x + sin(ta1) * p_y) * (a2 * cos(ta3) + a3);
double ta23 = atan2(fs23, sc23);
// 求解theta2 (弧度表示)
double ta2 = ta23 - ta3;
if (1(ta2 <= (0 + PIe) && ta2 >= -(PI/2 + 0.01))) {
    total = total - 1;
    continue;
}
// 求解theta4 (弧度表示)
double fs4[2] = { -a_x * sin(ta1) + a_y * cos(ta1), a_x *
sin(ta1) - a_y * cos(ta1) };
double sc4[2] = { -a_x * cos(ta1) * cos(ta23) - a_y *
sin(ta1) * cos(ta23) + a_z * sin(ta23), a_x * cos(ta1) *
cos(ta23) + a_y * sin(ta1) * cos(ta23) - a_z * sin(ta23) };
double ta4 = 0;
if (fabs(fs4[k]) < 1e-3 && fabs(sc4[k]) < 1e-3) ta4 = theta_init[3];
else ta4 = atan2(fs4[k], sc4[k]);

```

```

// 求解theta5 (弧度表示)
double fs5 = -a_x * (cos(ta1) * cos(ta23) *
cos(ta4) + sin(ta1) * sin(ta4)) - a_y * (sin(ta1) *
cos(ta23) * cos(ta4) - cos(ta1) * sin(ta4)) + a_z *
(sin(ta23) * cos(ta4));
double sc5 = a_x * (-cos(ta1) * sin(ta23)) + a_y *
(-sin(ta1) * sin(ta23)) + a_z * (-cos(ta23));
double ta5 = atan2(fs5, sc5);
if (!(ta5 <= (PI + PIe) / 2 && ta5 >= -(PI + PIe) / 2)) {
    total = total - 1;
    continue;
}
// 求解theta6 (弧度表示)
double fs6 =
-n_x * (cos(ta1) * cos(ta23) * sin(ta4) - sin(ta1) *
cos(ta4)) - n_y * (sin(ta1) * cos(ta23) *
sin(ta4) + cos(ta1) * cos(ta4)) + n_z *
(sin(ta23) * sin(ta4));
double sc6 = n_x * (cos(ta1) * cos(ta23) * cos(ta4) +
sin(ta1) * sin(ta4)) * cos(ta5) - n_x *
cos(ta1) * sin(ta23) * sin(ta5) + n_y *
(sin(ta1) * cos(ta23) * cos(ta4) + cos(ta1) *
sin(ta4)) * cos(ta5) - n_y * sin(ta1) * sin(ta23) *
sin(ta5) - n_z * (sin(ta23) * cos(ta4) *
cos(ta5) + cos(ta23) * sin(ta5));
double ta6 = atan2(fs6, sc6);
// 保存结果
m = m + 1;
Theta[m][0] = ta1 ;
Theta[m][1] = ta2 ;
Theta[m][2] = ta3 ;
Theta[m][3] = ta4 ;
Theta[m][4] = ta5 ;
Theta[m][5] = ta6 ;
}
}
}

```

```

// 计算Theta中每组解与theta_init的欧几里得距离
double dist[8]; // 定义一个数组, 存储距离
double sum; // 定义求和变量
for (i = 0; i <= m; i++) { // 遍历Theta的每一行
    sum = 0; // 初始化求和变量为0
    for (j = 0; j < 6; j++) { // 遍历Theta的每一列
        sum += (Theta[i][j] - theta_init[j]) * (Theta[i][j] - theta_init[j]);
    }
    dist[i] = sqrt(sum);
}

// 找出距离最小的那组解的索引
int index = 0; // 定义索引变量, 初始化为0
double min_dist = dist[0]; // 定义最小距离变量, 初始化为第一个元素
for (i = 0; i <= m; i++) { // 遍历剩余的距离
    if (dist[i] < min_dist) { // 如果找到更小的距离
        min_dist = dist[i]; // 更新最小距离变量
        index = i; // 更新索引变量
    }
}

// 判断是否找到最合适的解
if (min_dist < threshold_err * arm_speed) { // 如果距离小于阈值乘以速度
    // 定义一个新的初始角度数组
    double theta_init_new[6];
    // 将Theta中最小距离对应的解赋值给新的初始角度数组
    for (j = 0; j < 6; j++) {
        theta_init_new[j] = Theta[index][j];
    }
    // 判断是否有异常情况
    if (total == 0) {
        for (i = 0; i < 6; i++) {
            theta_init_new[i] = theta_true[i];
        }
        //printf("运行异常: total为0\n");
    }
}

else if (max_angle1(theta_init_new)) {
    for (i = 0; i < 6; i++) {
        theta_init_new[i] = theta_true[i];
    }
    //printf("运行异常: theta_init_new中有元素的绝对值大于等于3*pi\n");
}
else { // 否则, 正常运行

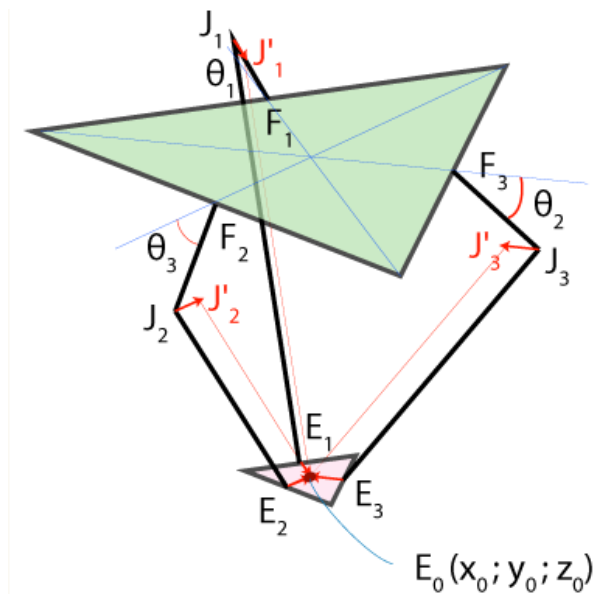
    for (i = 0; i < 6; i++) {
        theta_init_new[i] = theta_init_new0[i];
    }
    return;
}
}
else { // 否则, 距离过大
    for (i = 0; i < 6; i++) {
        theta_init_new[i] = theta_true[i];
    }
    //printf("运行异常: 距离过大\n");
}
}
}

```

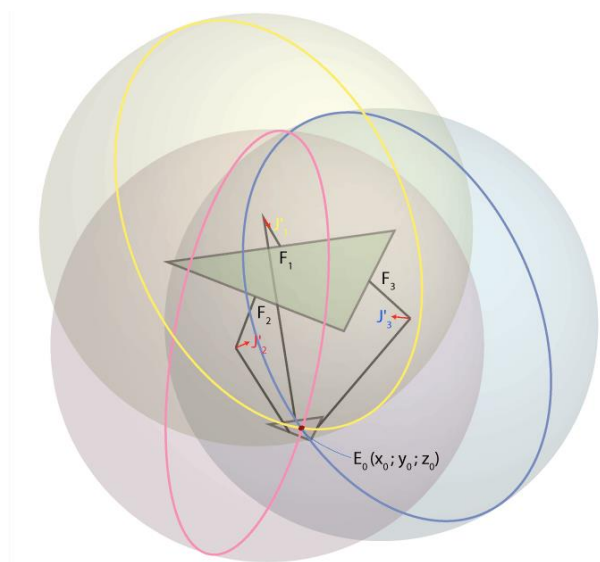
2.4.2.4 Delta 型机械臂解算

正运动学

输入 θ_1 、 θ_2 、 θ_3 ，输出到 x_0 、 y_0 、 z_0



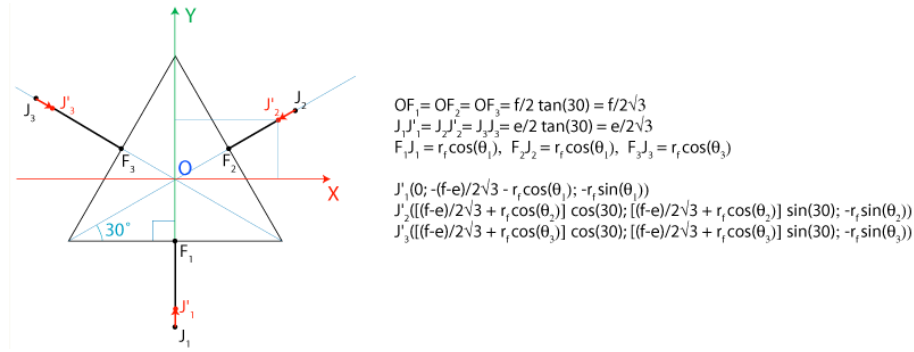
以 J_1 、 J_2 、 J_3 为球心， $J_i E_i$ 为半径，可以形成三个球面。



将球的中心从原中心移到 J'_i ，移动方向为沿着 $E_i E_0$ ，经过移动后，所有的球面将交于 E_0 点，

所以，为了找到 x_0 、 y_0 、 z_0 ，我们需要解三个二元三次方程组，形如：

$z_j^2 = r_e^2$, 首先, 找到新球心的坐标。



J_1, J_2, J_3 的坐标为 $(x_1, y_1, z_1), (x_2, y_2, z_2)$ and (x_3, y_3, z_3)

$$\begin{cases} x^2 + (y - y_1)^2 + (z - z_1)^2 = r_e^2 \\ (x - x_2)^2 + (y - y_2)^2 + (z - z_2)^2 = r_e^2 \\ (x - x_3)^2 + (y - y_3)^2 + (z - z_3)^2 = r_e^2 \end{cases} \Rightarrow \begin{cases} x^2 + y^2 + z^2 - 2y_1y - 2z_1z = r_e^2 - y_1^2 - z_1^2 & (1) \\ x^2 + y^2 + z^2 - 2x_2x - 2y_2y - 2z_2z = r_e^2 - x_2^2 - y_2^2 - z_2^2 & (2) \\ x^2 + y^2 + z^2 - 2x_3x - 2y_3y - 2z_3z = r_e^2 - x_3^2 - y_3^2 - z_3^2 & (3) \end{cases}$$

$$w_1 = x_1^2 + y_1^2 + z_1^2$$

$$\begin{cases} x_2x + (y_1 - y_2)y + (z_1 - z_2)z = (w_1 - w_2)/2 & (4) = (1) - (2) \\ x_3x + (y_1 - y_3)y + (z_1 - z_3)z = (w_1 - w_3)/2 & (5) = (1) - (3) \\ (x_2 - x_3)x + (y_2 - y_3)y + (z_2 - z_3)z = (w_2 - w_3)/2 & (6) = (2) - (3) \end{cases}$$

From (4)-(5):

$$x = a_1z + b_1 \quad (7)$$

$$y = a_2z + b_2 \quad (8)$$

$$a_1 = \frac{1}{d}[(z_2 - z_1)(y_3 - y_1) - (z_3 - z_1)(y_2 - y_1)] \quad a_2 = -\frac{1}{d}[(z_2 - z_1)x_3 - (z_3 - z_1)x_2]$$

$$b_1 = -\frac{1}{2d}[(w_2 - w_1)(y_3 - y_1) - (w_3 - w_1)(y_2 - y_1)] \quad b_2 = \frac{1}{2d}[(w_2 - w_1)x_3 - (w_3 - w_1)x_2]$$

$$d = (y_2 - y_1)x_3 - (y_3 - y_1)x_2$$

Now we can substitute (7) and (8) in (1):

$$(a_1^2 + a_2^2 + 1)z^2 + 2(a_1(b_2 - y_1) - z_1)z + (b_1^2 + (b_2 - y_1)^2 + z_1^2 - r_e^2) = 0$$

最后, 我们要求解这个二次方程并找到 z_0 (我们应该选择最小的负方程根), 然后根

据等式 (7) 和 (8) 计算 x_0 和 y_0 。

自定义控制器端正运动学代码:

```

int RC::delta_calcForward(double theta1, double theta2, double theta3)
{
    float t = (f - e) * tan30 / 2;
    float dtr = pi / (float)180.0;

    theta1 *= dtr;
    theta2 *= dtr;
    theta3 *= dtr;

    float y1 = -(t + rf * cos(theta1));
    float z1 = -rf * sin(theta1);

    float y2 = (t + rf * cos(theta2)) * sin30;
    float x2 = y2 * tan60;
    float z2 = -rf * sin(theta2);

    float y3 = (t + rf * cos(theta3)) * sin30;
    float x3 = -y3 * tan60;
    float z3 = -rf * sin(theta3);

    float dnm = (y2 - y1) * x3 - (y3 - y1) * x2;

    float w1 = y1 * y1 + z1 * z1;
    float w2 = x2 * x2 + y2 * y2 + z2 * z2;
    float w3 = x3 * x3 + y3 * y3 + z3 * z3;

    // x = (a1*z + b1)/dnm
    float a1 = (z2 - z1) * (y3 - y1) - (z3 - z1) * (y2 - y1);
    float b1 = -((w2 - w1) * (y3 - y1) - (w3 - w1) * (y2 - y1)) / 2.0;

    // y = (a2*z + b2)/dnm;
    float a2 = -(z2 - z1) * x3 + (z3 - z1) * x2;
    float b2 = ((w2 - w1) * x3 - (w3 - w1) * x2) / 2.0;

    // a*z^2 + b*z + c = 0
    float a = a1 * a1 + a2 * a2 + dnm * dnm;
    float b = 2 * (a1 * b1 + a2 * (b2 - y1 * dnm) - z1 * dnm * dnm);
    float c = (b2 - y1 * dnm) * (b2 - y1 * dnm) + b1 * b1 + dnm * dnm * (z1 * z1 - re * re);

    // discriminant
    float d = b * b - (float)4.0 * a * c;
    if (d < 0) return -1; // non-existing point

    x = -(float)0.5 * (b + sqrt(d)) / a;
    y = (a1 * x + b1) / dnm;
    z = (a2 * x + b2) / dnm;
    x += 96.83;
    return 0;
}

```

3. 研发迭代过程

3.1 测试记录

3.1.1 机械臂

3.1.1.1 第一轴关节电机工作测试

测试目标：采用电机 AK70-10 正常驱动使用第一轴关节。

测试结果：电机使用异常，无法做到和其他电机联动。

结果分析：没有绝对值编码器；通信方面，电机上电需要比单片机早上电 10s，不然 can 线无法传输数据。

解决方法：更换达妙 8006 电机，这样，整个 6 轴机械臂用的都是达妙电机，开发生态完善，通信协议简洁。

3.1.1.2 第二轴关节电机工作测试

测试目标：测试第二轴关节电机工作稳定性。

测试流程：末端携带矿石将机械臂伸直保持姿态。

- 第一代未加重力补偿测试结果：

电机输出扭矩在 25-30N·m，电机发烫严重。

测试结果分析：经 2.1.3.2 分析，机械臂机构所需的最大扭矩会达到 42.9N·m。单靠关节电机的驱动是没办法满足机械臂工作的扭矩。

- 第二代加入重力补偿测试结果：

电机输出扭矩在 $10\text{N}\cdot\text{m}$ - $13\text{N}\cdot\text{m}$ ，电机发烫现象消失。

测试结果分析：压簧重力补偿机构的加入，贡献了大部分所需的扭矩，将扭矩转移给机械结构来承担。

3.1.1.3 不同等级兑换测试

测试目标：测试机械臂完成 1-5 级兑换的能力。

测试流程：1-5 级兑换各随机摆 5 个位姿，尝试利用机械臂完成兑换。

测试结果：能完成 1-4 级的全部位姿，5 级的部分位姿可能达不到。

测试结果分析：仿真和设计时没有考虑到机械臂的宽度问题影响程度，实际的机械臂在机械上的限位导致第五轴关节电机的俯仰角不够（见下图），造成有部分区域没办法到达，一些位姿没办法获得。



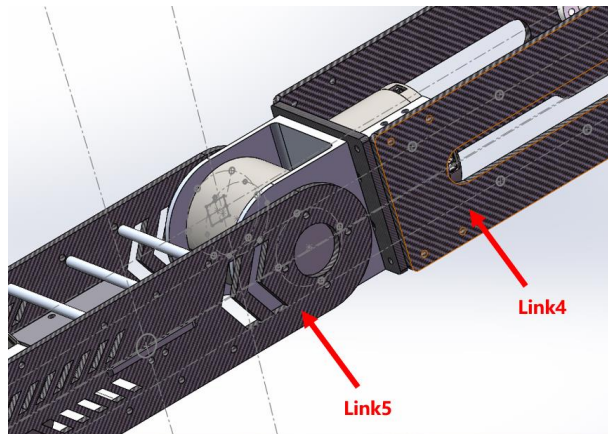
运动角度限制

3.1.1.4 第 5 级兑换补充测试

背景：测试发现 5 级兑换的一些机械臂没能达到的位姿。这是由于第五轴的折叠角度过小，在此基础上，我们思考如何改动最小得解决这个问题。

可能的解决方法有二：

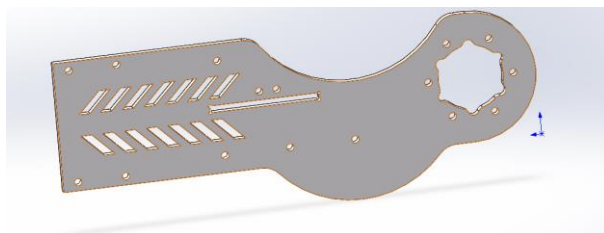
1. 削减 Link4 的宽度；
2. 弯曲 Link5，为折叠贡献空间



Link4、5 示意图

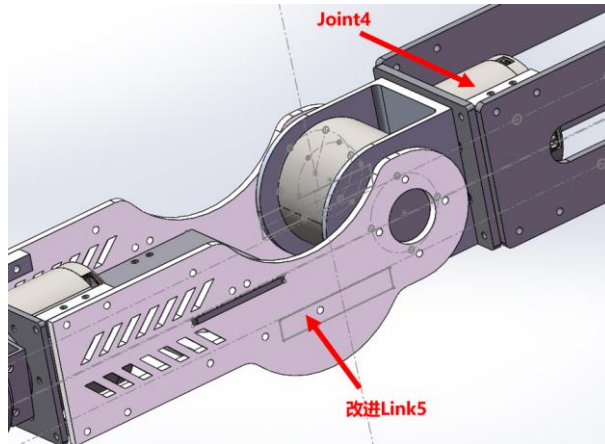
对于第一个问题，若削减 Link4 的宽度，会造成末端机械臂大，Link4 的刚度问题可能会影响机械臂的工作。

对于第二个问题，我们将 Link5 改为了弯曲型（见下图）。这样子的改动可以让向上摆动的角度变大，但是这样削减了向下摆动的范围。



修改后的 Link5 示意图

但是在软件上进行限位可以避免无法下摆的问题。当需要下摆时，Joint4 可以旋转 180° 进行下摆姿态的贡献。



修改后的 Link5 示意图

测试目标：机械臂在改进 Link5 后有完成 5 级兑换的能力。

测试流程：5 级兑换随机摆 5 个位姿，尝试利用机械臂完成兑换。

测试结果：能完成 5 级的全部位姿。

3.1.2 取矿测试

3.1.2.1 小资源岛

测试目标：用机械臂取到小资源岛的矿石。

测试流程：在自制的小资源岛模型放置矿石，用机械臂取若干次，统计成功率。

测试结果：成功率为 70%。

结果分析：

当从侧面取矿时，为了让吸盘吸紧矿石，需要施加一定的正压力。然而，在吸盘吸紧矿

石后，侧向压力会导致其与小资源岛的挡板发生摩擦自锁，使得矿石无法被拿起。



矿石在拿起过程中变倾斜，发生自锁

解决方案：在末端前侧增加一个吸盘，这个吸盘专门负责取银矿。这样能够从上方拿出矿石，也不影响原来的吸盘。



加入第二个吸盘

测试目标：测试从上方取矿的成功率。

测试流程：在自制的小资源岛模型放置矿石，用机械臂取若干次，统计成功率和问题。

测试结果：成功率 100%，从上方可以很轻松的取出矿石。

3.1.2.2 大资源岛

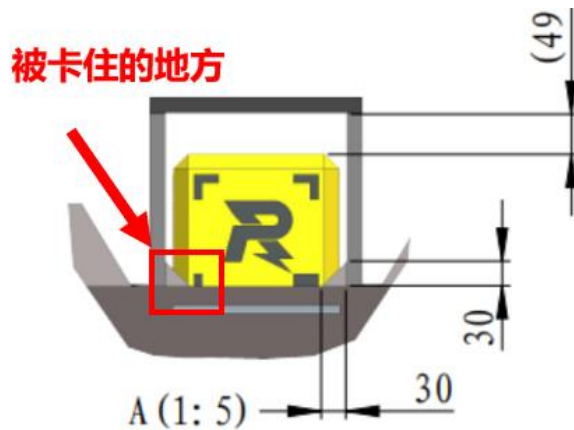
测试目标：用机械臂取到大资源岛的矿石。

测试流程：在自制的大资源岛模型放置矿石，用机械臂取若干次，统计成功率。

测试结果：成功率为 80%。

结果分析：

失败的原因都是因为操作手操作不当，吸取矿石准备退出坑道时被护肩卡住导致矿石掉落。



大资源岛矿石坑道示意图

解决方案：

计算好取完金矿石应抬高到的 z 轴高度，写出一套自动化流程进行取矿。

3.1.3 自定义控制器测试

测试目标：用 Delta 型自定义控制器控制机械臂完成 1-5 级兑换的能力。

测试流程：1-5 级兑换各随机摆 5 个位姿，尝试利用 Delta 型自定义控制器机械臂完成兑换。

测试结果：兑换平均时间在 40s。

结果分析：Delta 型自定义控制器臂长太短，空间姿态解算映射比例过小，灵敏度高，操作难度大。

解决方案：增加 Delta 型自定义控制器的臂长。

3.2 版本迭代过程记录

版本号或阶段	功能或性能详细说明	完成时间
机械臂 V1.0	功能： 实现设想的各种控制模式，能够手动操控或自动完成空间轨迹的跟随； 实现 1-4 级兑换，以及部分情况的 5 级兑换。 性能： 长时间工作时电机发烫严重； 第一轴电机用 AK70-10 需比单片机先上电，赛场无法保证。	2023.12.25
机械臂 V1.1	功能： 正常上电正常工作；	2024.1.4
机械臂 V2.0	功能： 更改 Link5 的形状，能完成 5 级的所有位姿兑换。	2024.1.15
机械臂 V3.0	功能： 加入重力补偿机构。 性能： 第二轴电机发烫现象消失。	2024.2.5
末端执行器 V1.0	功能： 实现一个吸盘的支撑，能够吸取矿石。	2024.1.4
末端执行器 V2.0	功能： 实现两个吸盘的支撑，能够吸取矿石；	2024.3.15

版本号或阶段	功能或性能详细说明	完成时间
	偏置一个吸盘，完成对小资源岛矿石的吸取。	
存矿机构 V1.0	功能： 实现两个矿石的储存。	2024.2.20
工程机器人 V1.0	功能： 兑换 1-4 级矿石，以及部分 5 级情况的兑换； 储存 2 个矿石； 全向移动。	2024.2.25
工程机器人 V2.0	功能： 兑换 1-5 级矿石； 储存 2 个矿石； 机械臂工作时断电，机械臂会弹到竖直位置。保证机构断电不会坠地； 全向移动。	2024.3.25
自定义控制器 V1.0	功能： 实现 RS232 通信； 摇杆控制。	2024.2.2
自定义控制器 V2.0	功能： 实现 RS232 通信； Delta 型机械臂正运动学解算输出位姿。	2024.3.2

3.3 重点问题解决记录

序号	问题描述	问题产生原因	问题解决方案 &实际解决效果	机器人版本号或阶段	解决人员
1	1、第一轴电机 AK70-10 Can 线通信异常； 2、每次上电需要手动摆到初始角度。	1、电机需要先比单片机先上电 10s 进行初始化；	更换成达妙 8006 电机。	机械臂 V1.1	机械工程师：罗聪、江晨峰

序号	问题描述	问题产生原因	问题解决方案 &实际解决效果	机器人版本号或阶段	解决人员
		2、AK70-10 电机没有绝对值编码器。			嵌入式软件工程师： 陈豪威、唐佩洁
2	机械臂 5 级兑换有部分位姿不能到达。	第五轴电机的俯仰角被限制。	改变 Link5 的形状，为俯仰角度留空间。	机械臂 V2.0	机械工程师：罗聪 嵌入式软件工程师： 陈豪威
3	机械臂第二轴电机工作时间长会发烫。	第二轴输出扭矩在 $20\text{N}\cdot\text{m}$ - $30\text{N}\cdot\text{m}$ 。	添加重力补偿机构。	机械臂 V3.0	机械工程师：罗聪、 江晨峰 嵌入式软件工程师： 陈豪威、唐佩洁
4	从侧面取小资源岛矿石成功率低。	侧面吸盘的正压力会使矿石在小资源岛的矿坑里发生自锁，无法取出	偏置一个吸盘从上方取出矿石。	末端执行器 V2.0	机械工程师：罗聪、 江晨峰 嵌入式软件工程师： 陈豪威、唐佩洁
5	Delta 型自定义控制器操作难度大。	Delta 型自定义控制器臂长太短，空间姿态解算映射比例过小。	增加杆长。	自定义控制器 V2.0	机械工程师：熊毅超 嵌入式软件工程师： 陈豪威、唐佩洁

4. 团队成员贡献

姓名	基本信息 (专业、年级、队内角色)	主要负责工作内容描述	贡献度 (所有成员贡献度合计为 100%)
罗聪	机械设计制造及其自动化、大三、工程组组长&机械结构设计负责人	负责工程机器人整体的功能规划和项目安排； 负责机械臂 1-6 关节的设计	25%
陈豪威	通信、大三、软件&硬件开发负责人	负责工程机器人硬件和嵌入式开发；	25%
江晨峰	机械设计制造及其自动化、大二、机械组成员	负责底盘的结构设计	15%
唐佩洁	通信、大三、电控组成员	负责工程机器人硬件和嵌入式开发；	15%
熊毅超	工科试验班、大一、机械组成员	负责自定义控制器的结构设计	10%
林灿豪	电子信息科学与技术、大二、电控组成员	负责自定义控制器的算法开发	10%

5. 参考文献

- [1] D. Lee and T. Seo, "Lightweight Multi-DOF Manipulator With Wire-Driven Gravity Compensation Mechanism," in IEEE/ASME Transactions on Mechatronics, vol. 22, no. 3, pp. 1308-1314, June 2017, doi: 10.1109/TMECH.2017.2681102.
- [2] RM2023- 南方科技大学 ARTINX 机器人队 - 工程机器人 - 机械结构开源 ,
<https://bbs.robomaster.com/forum.php?mod=viewthread&tid=22759>
- [3] RM2023- 上海交通大学 云汉交龙战队 - 工程机器人 - 机械结构开源,
<https://bbs.robomaster.com/forum.php?mod=viewthread&tid=22758>
- [4] RM2023- 浙江大学 Hello World 战队 - 工程机器人 - 机械结构开源,
<https://bbs.robomaster.com/forum.php?mod=viewthread&tid=22818>
- [4] [美]John J. Craig. 机器人学导论[M]. 负超、王伟. 北京: 机械工业出版社, 2021

6. 技术方案复盘

6.1 赛场性能表现情况分析

6.1.1 底盘

经过三场 BO2 的总共 6 小局比赛, 轮组的固定方案是合理的, 没有出现掉轮卡轮的情况。

6.1.2 机械臂

机械臂的重力补偿表现是符合预期的, 在被击杀后, 机械臂也没有出现整个砸倒的情况, 唯一需要注意的是, 在进行了一天的调试和比赛后需要进行张紧的加固, 但是本结构有考虑到这点, 张紧的过程也并不复杂。

6.1.3 自定义控制器

在赛场上, 自定义控制器的解算控制还是没有出现断连的情况, 但是存在固定不牢导致不稳的情况。

6.2 赛场性能表现与规则对比分析

六轴机械臂的重力补偿很符合规则的击杀断电处理, 能够很好地进行机构的保护。

6.3 经验总结

六轴机械臂的向前伸展尺寸还可以做的更极限一点, 并且结构上应考虑末端的轻量化, 这样能实现在一个矿道抓取两个金矿石的要求。



邮箱: robomaster@dji.com

论坛: <http://bbs.robomaster.com>

官网: <http://www.robomaster.com>

电话: 0755-36383255 (周一至周五10:30-19:30)

地址: 广东省深圳市南山区西丽街道仙茶路与兴科路交叉口大疆天空之城T2 22F