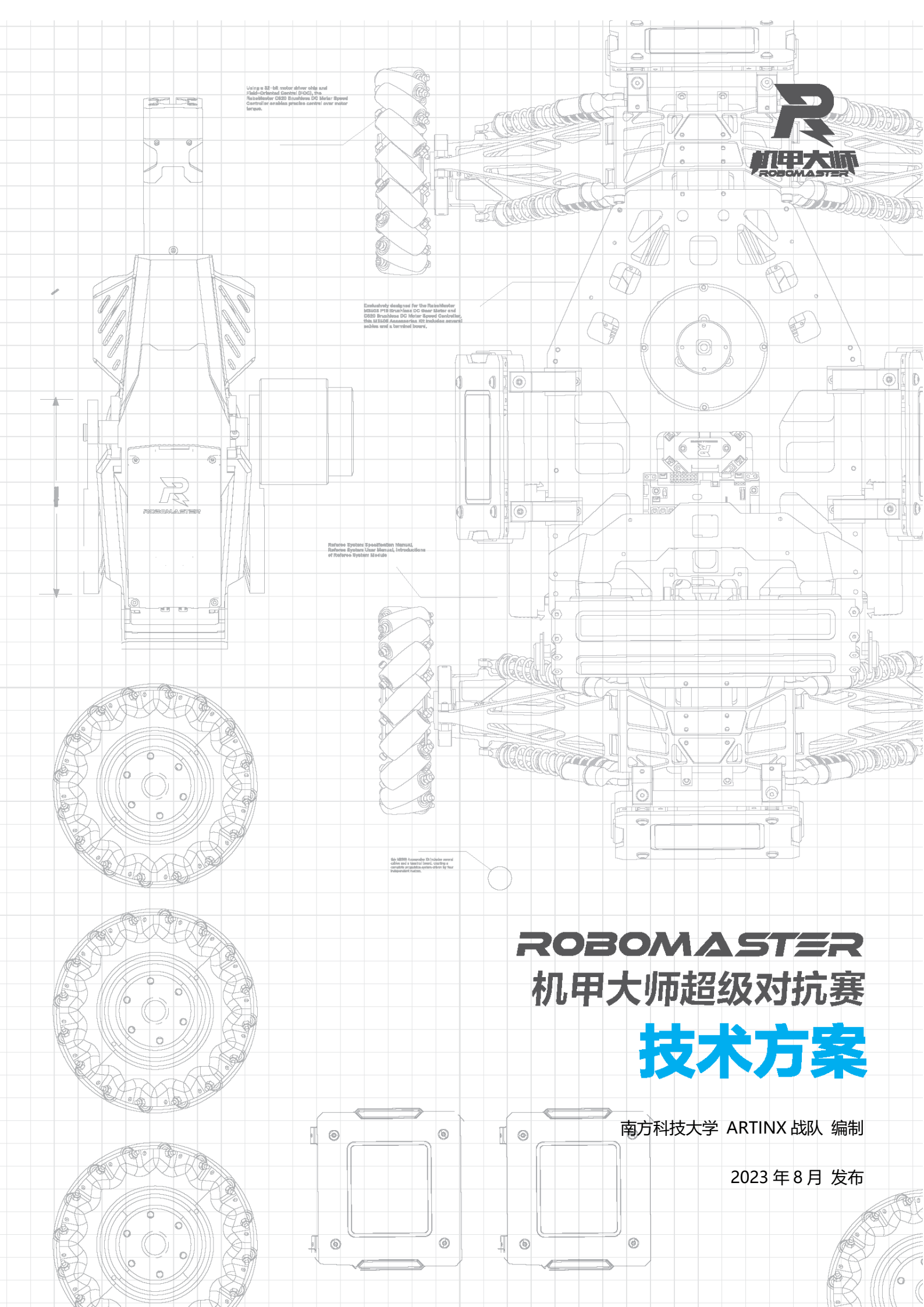


ROBOMASTER
机甲大师超级对抗赛
技术方案

南方科技大学 ARTINX 战队 编制

2023 年 8 月 发布

前言

本成本报告由南方科技大学 ARTINX 编制，适用于 RoboMaster 2023 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	李崇珊	张宸翰
硬件	盛李杰	
软件	盛李杰	
算法	刘家荣（已离队）	

目录

1. 概述

1.1 背景&目标

本项目是一台可搬运特定道具的移动机器人，面向 2023 赛季 RoboMaster 比赛（后简称 RM 比赛）的规则，将作为“工程机器人”参加比赛。在 RM 比赛中，工程机器人的主要任务是搬运场地道具来为己方获得优势，以及进行一些拖拽救援和战术阻挡。工程机器人是没有直接进攻能力的辅助型的机器人，但是它的辅助功能非常重要，能够很大的左右比赛的局势。

在 2023 赛季的 RM 比赛规则（v1.0 版本）中，与工程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。关于规则的解读和一些数据的计算已在赛季规划中详细分点写出，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
- 金银矿金币价值差距小，银矿的性价比比金矿更高
- 工程仍然具有重要的救援和战术阻挡的职能

我们认为工程车的战术职能的优先级为：提供经济>救援>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，储存矿石和拖拽救援的功能。特别的，在获取经济的方面，我们认为 23 赛季中银矿比金矿更重要，如果 5 个银矿（v1.0 版本规则）能全部兑换，价值会高于在重要资源岛的金矿抢夺中获得优势。

综合考虑下，赛季初我们决定放弃金矿，只取银矿。我们在 23 赛季的战术目标是，在 7 分钟的比赛中，前 4 分钟内取完并兑换完全部银矿，完成后视情况进行拖拽救援和在团战中进行战术阻挡。

在分区赛后，经历了规则的改动，对工程来说主要的两个改动是血量减半和银矿数量减少。前者会影响整体战术，进而影响工程机器人的功能需求；后者直接对具体功能的实现产生影响。在规则改变后，我们原本计划的“一次性带走 5 个银矿”相关的研发全部报废，因为整体功能的取舍也跟这个息息相关，所以整体方案在规则改变后也变的非常小丑（不取金矿等策略，是基于第一版规则的工程车的战术地位的，到了国赛取不了金矿就显得非常小丑）。但是考虑到国赛前研发时间有限，以及我们人手有限，最后的决策是**不改变整体方案**。

1.2 其它学校机器人分析综述

1.2.1 RMUC 21 赛季综述

（叠甲：以下对一些队伍机器人的功能和评价来自当年在赛场上的印象和一些比赛的视频，难免有些错误的地方。如有疏漏或出入，请以实际情况为准。）

21 赛季开始，砍掉了工程的基础尺寸，引入经济系统，进入了到目前的这一版大规则。在 21 赛季，绝大多数队伍都采取了之前赛季祖传的 2 轴/3 轴平动平台+夹爪的方案。其中以上海交通大学为代表，3 轴平台+夹爪方案，实现了稳定的取矿兑换，以及空接的功能。除了主流的平动平台+夹爪的方案，也有以一些队伍为代表的队伍采用了一些独特且有所成效的方案，包括但不限于：

学校 **e.g. 上海交通大学**	**方案** **e.g. 3 轴平动平台+夹爪**
东北大学	2 轴平台+吸盘
哈尔滨工业大学	特殊四连杆+夹爪
深圳大学	抬升+转头+夹爪 （转头指旋转整个夹爪机构，相当于用一个旋转副取代了 2 轴平台的横向的平移副）
广东工业大学	机械臂+吸盘

以上的这些队伍方案独特的同时，在国赛中表现出了稳定的性能，给大家留下了深刻的印象。

东北大学使用了真空吸盘作为末端执行器，吸盘的优秀表现一举打破了旧时代夹爪取矿的统治地位，后来许多队伍也开始开始采用吸盘，这都要感谢东北大学 21 赛季在赛场上的精彩表现。

哈尔滨工业大学采用了一套基于四连杆的很特殊的机构，其巧妙的设计让大家连连赞叹。笔者同样认为设计者设计的非常巧妙，但是笔者也认为实际上（包括后面 22 年出现的其他）四连杆的方案在空间占用和稳定性上没有明显的高于平动平台机构的优势；同时也没有摆脱传统夹爪的束缚。

深圳大学的创新原理简单，但效果稳定。用很简单的方法实现了平动机构的平替，平淡而具有启发性。（我们今年的 SCARA 构型机械臂也与这种机构有相似之处）

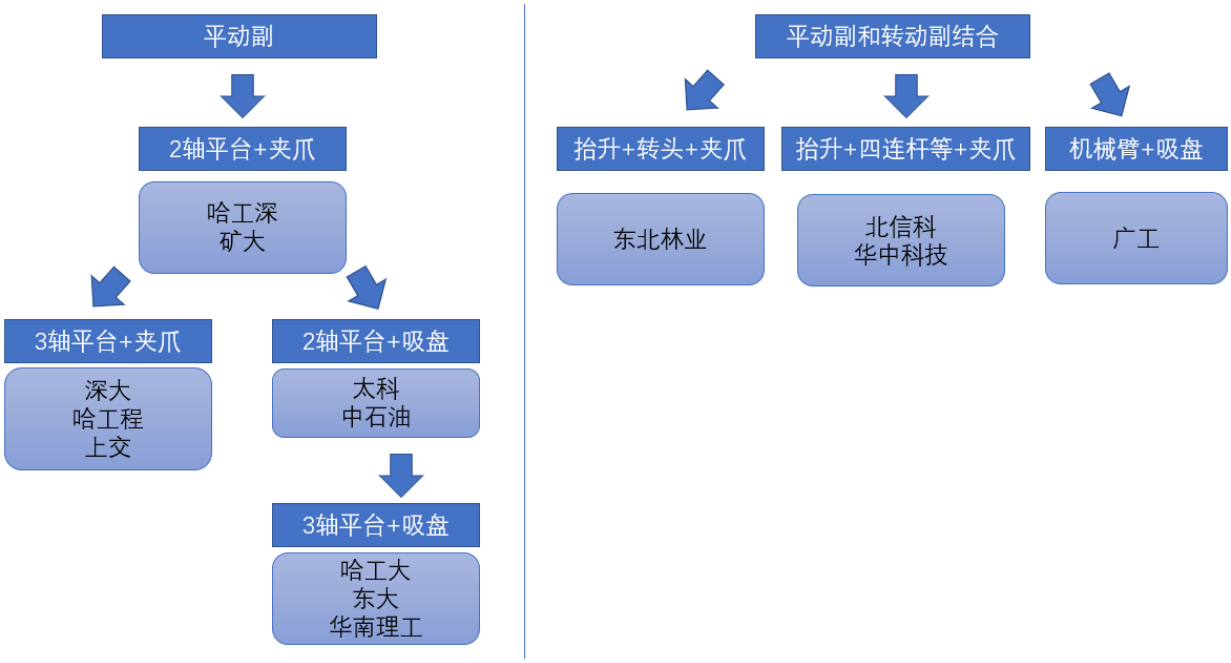
广东工业大学更是重量级，在 21 赛季做出了纯旋转关节（不算末端执行器）的机械臂。广东工业大学在 21 赛季虽然进入了全国赛，但工程机器人的表现说不上是十分的稳定（赛场表现看着感觉当年并没

有上解算和路径规划？）。尽管如此，这样大胆的尝试着实是让人眼前一亮，在谁也想不到后续规则走向的 21 赛季，着实是让人产生油然的尊敬。

以上是主要构型的概括，除此之外有许多队伍做出了储矿的功能，方案多种多样。因为不是主要功能，这里不加赘述。印象中一些更加进阶的功能（如矿石姿态调整和拿障碍块）实现的队伍屈指可数。（年代久远，个人只对浙纺的取障碍块比较有印象了）

1.2.2 RMUC 22 赛季综述

22 赛季，参赛队伍的方案的多样性逐渐发展，主要构型上，大部分队伍还是采用传统平动平台+夹爪的方案，但采用非传统方案的学校较之前增加了许多。其中，在功能上，如果我们简单的把 3 轴平台看成 2 轴平台的上位替换，吸盘看成夹爪的上位替换的话，大体的方案分布如图所示：



（只记录了功能较稳定的部分学校）

22 赛季各种的百家争鸣，其中把取矿兑矿空接等功能实现的很稳定的学校有很多，且各种不同构型方案都有能稳定实现这些功能。相比 21 赛季 22 赛季队伍的整体水平提高，能把最基本的取矿和兑换做稳定的队伍非常多。在 22 赛季（国赛前规则）的环境下，**不同构型的选择对功能的影响逐渐减小，只要基本的机械结构设计和电气控制能做好，任一种构型都能有不俗的表现**。

除此之外，有很多队伍在改变构型的同时实现了一些额外的功能，包括但不限于矿石姿态调整，单独取地面矿石的机构，取障碍块机构等。

1.2.3 RMUC 23 赛季综述

23 赛季，笔者作为参赛队员，备赛之余没有细细去整理各个队伍的构型，**以下是我作为参赛队员的一些偏主观的感受，不保真，请大家理性看待**。

分区赛阶段，在南部分区赛能看到小半数的队伍掏出了新研发的机械臂工程车，后面的中部和北部也有一些学校。在 22 赛季各个学校普遍已经把工程车招到相对稳定的情况下，许多学校有尝试去挑战做出一些新的东西。尽管整体看来机械臂方案工程车做到稳定的其实不太多，但笔者还是很感动有这么多人在尝试创新。

在复活赛-国赛阶段，大部分队伍还是回归了 3 轴位移平台+3 轴正交旋转机械臂的经典 plus 构型，基本上队伍也都能做到稳定的最高级兑换了。少部分学校仍然是机械臂方案。除此之外还有东北大学等学校在取矿流程上做更创新的尝试。从方案上限来看，机械臂方案工作空间就是小于位移平台的方案，具有原理性的缺点。剩下的这些使用纯机械臂的学校，基本上都是有额外功能考虑的学校，如我们（南科大）本来想储 4 个矿（被砍后成为小丑），上海交通大学的能小陀螺等。笔者认为上交和东大的工程非常有启发性，脱离了传统 3+3 构型的束缚，工程车其实可以有更高的功能上限。当然，这也是建立在队伍的工程技术积累上的，基本功足够扎实，才能挑战新的东西。

总的来说，整体来看大部分学校已经能够做到稳定最高级兑换。纯机械臂方案的学校稳定性普遍还是不如 3+3 经典构型，各学校的机械臂的相关技术看来还在发展期。一些学校在整体方案思路上有创新，在实现基本功能外实现了新的创新（东大，上交等）。工程机器人的功能上限还是有很多发挥的空间。

1.3 机器人功能定义

1.3.1 核心功能取舍

1.3.1.1 储矿

如前文所述，我们的目标战术为：开局直奔银矿，尽量在 3-4 分钟内把所有银矿换完，期间队友死了直接买活，兑完后工程车出去看情况进行救援或者参加对抗。同时，为了使收益最大化，我们需要尽量挑战高等级的兑换站模式。

23 赛季的小资源岛位置发生变化，从启动区正面移动到了稍远一点的地方，使得小资源岛到兑换站的距离变得更加远一些。同时这条路会经过哨兵巡逻区，工程可能需要对地面的哨兵绕行。这些因素会导致工程取矿回来兑矿一次的时间成本变高。

如果能减少工程在小资源岛-兑换站间移动的次数，可以缩短我们完成所有银矿兑换的任务的时间。因此我们希望能一次性储存更多的矿石。

因此我们希望储矿机构**能储存 4 个矿石**。这样我们的战术为开局直奔小资源岛，一次性取完所有五个银矿（储存 4 个+末端持有一个），然后移动到兑换站，一次性兑换完 5 个银矿，完成前期的经济供给任务。

在规则更改后，**需求更改为储存 2 个矿石**。但此时已经不打算更改整体方案了，以下的所有为了储矿做的努力（最大化减少取矿/兑换机构空间等），都是为了一开始的存储 4 个矿的目标。

1.3.1.2 取矿/兑换

为了保证储矿机构的空间，我们需要**尽可能的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，改变构型，改变具体机械结构等方式来减少初始空间。在空间减少的同时，该机构还是应该能稳定的完成取矿，兑换以及与储矿交互的任务。

1.3.1.3 自定义控制器

在我们赛季初的规划里，本来只规划了上视觉辅助兑矿，不打算研究自定义控制器，因为我们核心研发人员不足，不敢涉足。在分区赛后，我们看到了南航的位姿同步的自定义控制器的精彩表现，发现了两个事实：第一，从录像来看自定义控制器对兑换效率的提升是很高的，人可以同时控制 6 个自由度，且能实现一些“蹭进去”的操作；第二，自定义控制器可以用很低技术力的成品方案解决最难的问题（从录像上看南航用的也是 t265 相机），只需要攻克官方图传链路的通信即可，整体技术难度不大。

在国赛前两个月，队伍回归了一位有经验的电控/算法方向的老队员。在我们分区赛后的评估下自定义控制器的性价比非常高，此时又有合适的担当人员能够负责这个项目，于是这个项目在分区赛后开始启动。

1.3.2 整体功能定义

23 赛季工程机器人具体功能定义如下，红色为分区赛后修改或新增：

功能模块	功能
底盘	能够全向移动 上层的机构运动时保持底盘稳定，不移动，不翻车 能在平地，坡，起伏路段等地形完成救援 ● 操作手能在 2s 内完成己方车辆的固连
取矿/兑换机构	能够获取小资源岛的矿石 ● 30s 内完成 3 个矿石的获取 能够完成最高 4 级的兑换 能视觉识别兑换站位姿，辅助自定义控制器兑换 ● 30s 内完成单次兑换 系统稳定 ● 在各个模式间运行或切换时不发生逻辑错误 ● 在受到冲击后能回复原始状态，继续完成任务
储矿机构	能够放入和拿出矿石 能够储存 2 个矿石 在机器人的移动和颠簸等过程中，不丢失矿石
其他功能	车壳坚固，外观美观 可防止弹丸等进入车体内部引发事故 线材和硬件易于维护 各个机构模块化 ● 单个机构 1min 内可以整体从车上拆卸 增加图传自由度，用背面当作正面开车，避免机械臂受击

1.4 机器人核心参数

名称	参数
重量	35.04kg
收缩尺寸（长*宽*高）	590*590*585
伸展尺寸（长*宽*高）	1100*835*995
执行器数量	M3508 电机：8 M2006 电机：2 GO-M8010-6 电机：2 直线气缸：3 真空发生器+吸盘：1
最大移动速度	2.5m/s
最大矿石持有数	3
气瓶工作时储存气压	≤20KPa
气瓶容量	0.8L
工作气压	0.45Mpa
工作真空度	-80KPa

1.5 设计方案

1.5.1 机械结构设计

1.5.1.1 整体方案

方案综述

在上面其他学校的综述中，我们看到目前的工程车主要构型是非常多样的。概括来说，在 22 赛季的规则下（国赛前的规则），所有这些机构都可以概括为具有 3 个自由度的空间平移机构，**不同之处在于平移**

副和旋转副的数量和选择。经典的 2/3 轴平动平台属于纯平移副，抬升+转头和四连杆等方案则是平移副+旋转副，广工的一代机械臂则是纯旋转副。总而言之，这一整个位移机构可以看作一个**广义的机械臂**，由各个关节组成，这些关节可以是平移副，也可以是旋转副。不同构型的机械臂具有不同的工作空间，只要工作空间能满足我们功能的需求，即是可能合适的设计。

而机构的刚性和稳定性等则是依靠具体的机械机构设计和电气控制提升。没有“平动平台一定比（全是旋转副的）机械臂稳的说法”，只是说设计的不好的机械臂会没有设计的好的平动平台稳定而已。当然，在比赛的环境下，不同的构型具有不同的研发难度和成本，这个反而是更加重要的因素。

面对上文提到的功能需求，结合研发难度，时间和金钱成本，以及队内人员等因素，我们决定使用 **4 矿的储矿机构+带吸盘的六轴机械臂**为核心的方案（后被规则砍掉而变成 2 矿）。4 个矿石的储矿机构是我们达到战术目标的核心，六轴机械臂通过构型的选择可以节省空间的同时满足工作空间的需求，同时末端执行器选择吸盘，减小对解算影响的同时还具有质量轻，稳定性强等优点。

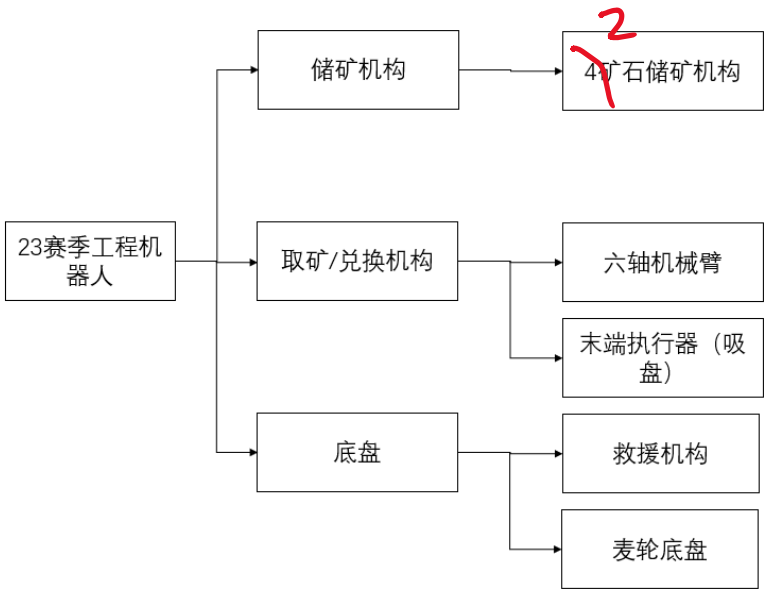
储矿机构部分，我们原计划研发 4 矿的储矿机构，后来为了国赛版规则，最后上了保底且更稳定的 2 矿储矿机构。2 矿储矿机构为横向储矿，车身后方有一个隔离的比 2 矿石稍大的空间，底部是一整根工业皮带，通过皮带的正向和方向运动来控制矿石在储矿空间中的运动。这部分原理非常简单，在实际测试的过程中功能也非常稳定。

机械臂部分，我们采用 **SCARA 构型+球型手腕的构型，**通过运动解耦的思想和一些策略，可以完成逆运动学的解算。相比 RRR 型机械臂，RRR 型的工作空间为球体，而 SCARA 构型的工作空间是一个圆柱，工作空间利用率更高（RM 的尺寸限制是立方体）；相比 PPP 型（3 轴平动平台），SCARA 构型平动关节少，机械臂本体的占用空间更小，因此可以给储矿机构留出更多空间。

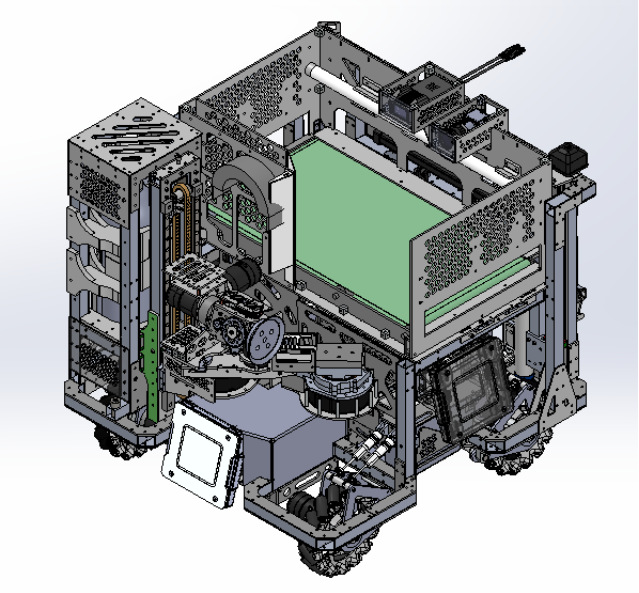
末端执行器部分，我们采用吸盘作为获取矿石的执行器，并且采用**真空发生器**作为负压的产生装置。吸盘相对夹爪质量更小，容错率也更高。真空发生器是由正压产生负压的装置，相比真空气泵能够避免震动的问题，因为我们的机器人保留了许多气缸执行器，具有正压气源，因此最后使用了真空发生器。再经过整机的实际测试后，最终还在末端执行器**增加了一个额外的旋转自由度**，后文会详细介绍。

除了以上的核心机构之外，还有底盘和救援机构的部分。对于底盘设计，考虑到取矿时最好需要全向移动功能，考虑仍然使用麦轮方案。实际测试发现，因为机器人本身质量较大，机械臂运动时并不会导致车体相对地面的移动。救援方案采用传统拖拽救援思路，搭载于底盘上。使用钩爪+单向门结构结合。本体通过一个气缸来进行伸出和释放，单向门的结构使得操作手的操作更加简单，避免了传统旋转钩爪操作时可能会把待救援机器人撞走的缺点。

综上所述，23 赛季我队的国赛版工程机器人具体方案如下：



功能模块	具体机构	详细方案
底盘	麦轮底盘	麦轮方案。
	救援机构	旋转钩爪+单向门机构。气缸触发，钩的部分采用单向门结构，有利于操作手操作。
取矿/兑换机构	六轴机械臂	前三轴采用非传统的 SCARA 构型（PRR，第一关节为 P），后三轴采用经典的正交球形手腕构型（RRR）。使用 SCARA 构型而不是传统 RRR 构型可以减少关节的最大扭矩，机械臂的自重由机架承担，而不用电机持续输出扭矩。通过一些策略，可以完成六轴逆运动学的解算。
	末端执行器	负压产生器选择用真空发生器，可避免气泵的震动对机械臂的影响；末端为一个单吸盘，加一个旋转自由度，使末端吸盘可以到达水平和垂直两种状态，水平状态用于兑换，垂直状态用于取矿。
储矿机构	2 矿石储矿机构	水平储存两个矿石，底部为传送带，可以控制矿石的水平位置运动。



工程机器人

整车工艺

零件加工工艺：

零件类型	加工工艺
玻纤板/碳纤板	铣削
铝方管	钻孔/切割/铣削
非标零件	铣削/车削/攻丝/线切割/钣金折弯/3d 打印（PLA/ABS）
亚克力板/PVC 板/木板	激光切割

结构连接工艺：

结构	工艺
框架/结构件	铝方管与玻纤板铆接
局部立体结构	3d 打印/板材插接+螺丝连接

1.5.1.2 六轴机械臂

需求分析

功能需求

六轴机械臂是我们工程车的核心功能机构，用于完成矿石获取，兑换和储存的功能。为了完成上述功能，首先机械臂的工作空间需要大于以上的目标空间。同时在我们的战术安排中，储矿的功能非常重要，而储存 4 个矿石需要占用车体的大部分空间，因此机械臂整体需要尽可能的压缩初始空间。除此之外，机械臂的结构还应该具有基本的刚性和稳定性，同时设计应模块化，方便后续维修，

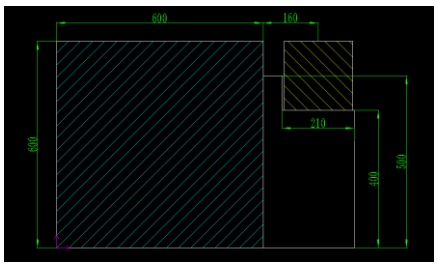
整理需求如下：

1. 工作空间大于目标空间
2. 初始占用空间尽量小
3. 连杆刚性足够，能够应对可能的冲击工况
4. 设计模块化，每个关节能单独拆卸

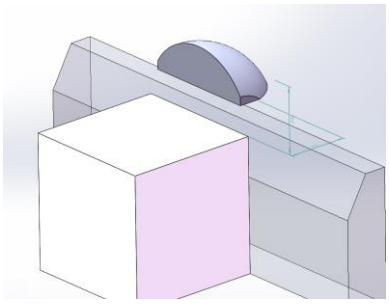
目标空间

我们的工程车的主要交互对象为小资源岛，兑换站和工程车自己的储矿机构。小资源岛和兑换站的目标空间示意图如下：

小资源岛：



兑换站（这里图只能展示出位置，其实目标空间还包括姿态）：

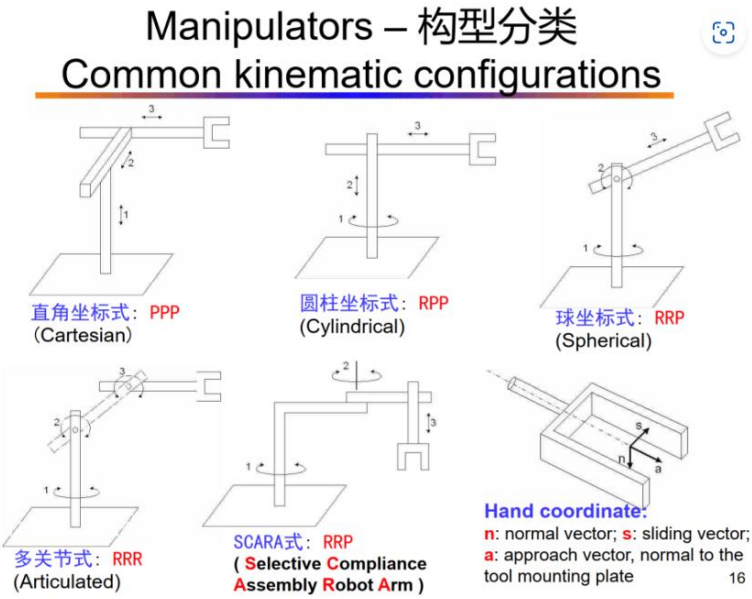


自身储矿机构的目标空间在车体内，且可以根据机械臂的构型调整位置，所以这里主要先考虑上面两个目标空间。

构型选择和比较

利用运动解耦的思想，把一台 6 轴机械臂（6 自由度）分成前 3 轴（位置三轴）和后 3 轴（姿态三轴），尽管最后逆运动学的解算不能完全解耦，但是可以通过一些策略使其能够相对独立的求解两个部分的关节角。为了解算方便，**末端姿态三轴直接采用经典的正交型球形手腕（RRR）**，而位置三轴的选择需要进一步讨论。

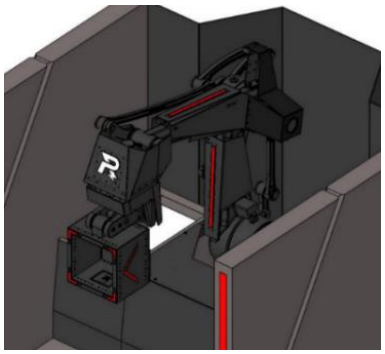
而前三轴有许多不同的构型，几种比较经典的机械臂构型如下：



PPP：经典平动三轴平台，其优点是可采用成熟的滑轨抬升机构（开源一大堆，我们队伍也有较为成熟的技术积累），运动较为稳定，且运动解算极为简单；缺点是平动机构浪费车体的初始空间大，整机重量也高居不下，重心高更易翻车。工作空间为矩形，针对规则的变形空间利用率最高。

经典 PPP 型机械臂（赞助商打钱）

RRR：经典构型，大多数工业机械臂采用此构型，网上能找到成熟的解算方案，也是大部分工业机器人采用的构型。但是 23 关节需要支持机械臂的重力，具有持续的高负载。工作空间为（半）球形，规则的变形空间利用率较低。

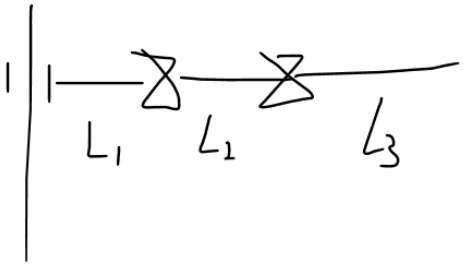


经典 6R 构型

RRP（SCARA 式）：经典机械臂构型，有两个旋转关节和一个平动关节，解算也较为方便，因为构型特殊，旋转关节不用负载机械臂的自重。但重力由机架承担，对旋转关节和连杆的刚度要求比较高。工作空间为圆柱形，规则的变形空间利用率低，但比 RRR 型高。

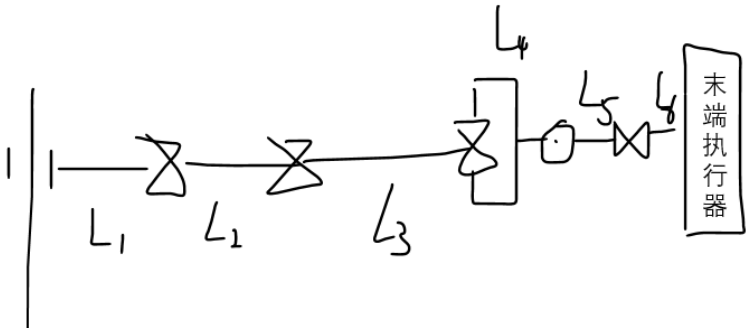
SCARA 型机械臂

为了给储矿机构节省空间，我们不希望采用 PPP 型机械臂构型；考虑到电机扭矩和工作空间，我们排除了 RRR 构型，最终选择了 RRP 的 SCARA 机械臂构型。同时，在经典的工业机器人中，SCARA 构型的平动关节为最后一关节；在我们的机器人上，我们让平动关节作为第一关节，以尽量减少后面关节的重量，可以提高响应速度和连杆稳定性。



第一关节为平动关节的 SCARA 式构型

加上末端三轴后，示意图如下：

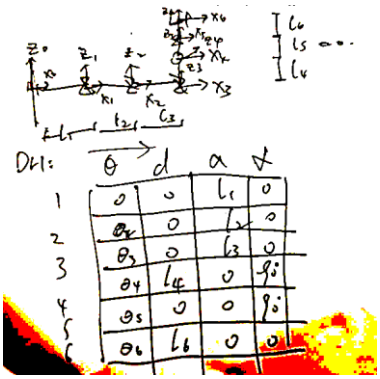


PRR（SCARA）+RRR（球形手腕）的六轴机械臂

仿真与参数确定

机械臂定义

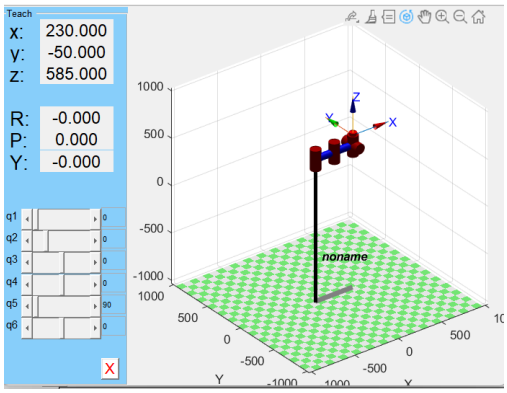
机械臂的坐标轴定义与 DH 矩阵如图：



坐标轴方向与 DH 参数（注意球形手腕：l5=0）

利用 robotic toolbox for matlab 对我们的机械臂进行初步的仿真，来确定关键的尺寸。

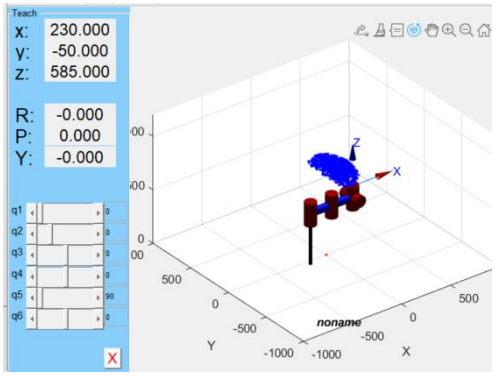
```
%% 生成机械臂
% 杆长 （这里的杆长已经是调整好的了）
l1=0;
l2=215;
l3=215;
l4=0; % 几个轴都有 z 偏置，在此视为 0，即一开始的基准平面是过 5 轴的转轴的平面
l5=0;
l6=100;
L=[l1,l2,l3,l4,l5,l6];

% DH 参数  [theta d a alpha]
L1=Link([0 0 L(1) 0]);
L1.jointtype='P';
L2=Link([0 0 L(2) 0]);
L3=Link([0 0 L(3) 0]);
L4=Link([0 L(4) 0 -pi/2]);
L5=Link([0 0 0 pi/2]);
L6=Link([0 L(6) 0 0]);

robot=SerialLink([L1 L2 L3 L4 L5 L6]);
base_bias=[-200,-50,485];              %%这里已经调整好了
robot.base=transl(base_bias);
view(3)
robot.teach;
```



我们的机械臂

工作空间

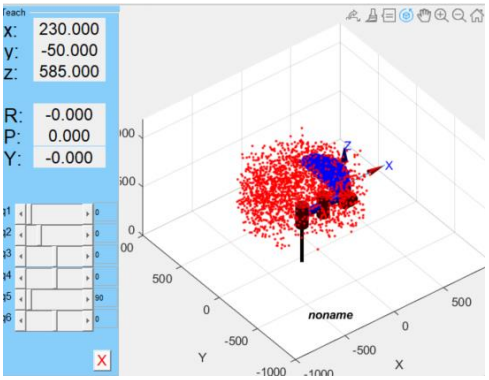
根据规则的数据，画出**兑换站**的目标空间：



蓝色为兑换站的工作空间

加上一些关节角限制，画出机械臂的工作空间：

```
%% 生成工作空间
num=2000;   %随机生成点的数量
P=zeros(num,3);

for i=1:num
    q1=robot.links(1).qlim(1)+rand*(robot.links(1).qlim(2)-robot.links(1).qlim(1));
    q2=robot.links(2).qlim(1)+rand*(robot.links(2).qlim(2)-robot.links(2).qlim(1));
    q3=robot.links(3).qlim(1)+rand*(robot.links(3).qlim(2)-robot.links(3).qlim(1));
    q4=robot.links(4).qlim(1)+rand*(robot.links(4).qlim(2)-robot.links(4).qlim(1));
    q5=robot.links(5).qlim(1)+rand*(robot.links(5).qlim(2)-robot.links(5).qlim(1));
    q6=robot.links(6).qlim(1)+rand*(robot.links(6).qlim(2)-robot.links(6).qlim(1));
    q=[q1,q2,q3,q4,q5,q6];
    T=robot.fkine(q);
    P(i,:)=transl(T);
end


plot3(P(:,1),P(:,2),P(:,3),'r.');
robot.plot([0,0,0,0,0,0],'tilesize',100);
    hold on
```



红色为工作空间

红色为工作空间，我们调整各杆长和基座相对原点的 bias，使红色空间完全覆盖蓝色空间，即为完成兑换站的目标。（上述的各杆长和基座的 bias 已经是我们调整好之后的数值了）

小资源岛的工作空间为 z500+的部分区域，较为简单，只要我们的第一关节（抬升机构）的行程能够达到这里，就能够完成取小资源岛矿石的任务。同时，23 关节的杆长调整至底盘静止时可以拿到 3 个银矿的长度。

可行性仿真

我们需要验证我们的机械臂是否一定能达到兑换站目标空间中的所有位姿（上面的工作空间只能看出位置，不能看出姿态），因此，我们需要进行兑换站所有位姿的可达性测试。

测试详见 1.6.1.1 中的测试记录，我们通过对随机生成数据的数据进行解算仿真，验证了我们的构型的六轴机械臂能够实现所有兑换站所需位姿。

结构设计

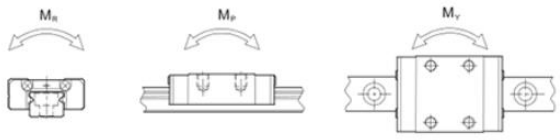
第 1 关节（平动关节）

1 关节是一个竖直方向的平动关节。往后的 5 个关节质量之和约 5kg，且可能存在冲击工况，这里采用了负载较大的链传动来实现抬升功能；同时通过两套微型直线导轨来保证直线运动的稳定性，同时增强结构的刚度。

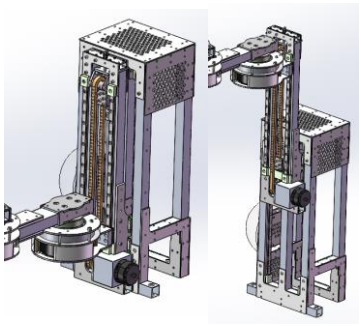
我们希望 z 轴覆盖从小资源岛的 500 到最大尺寸的 1000（实际上没有那么极限），因此我们希望这一关节的行程为 500（mm）。这里我们采用了两级同步抬升的机构，这样使压缩的尺寸能够在高 600 的范围内。

导轨和滑块我们使用了 MGN9 系列的产品，每一级为两根导轨并列，每根导轨上安置两个滑块，来避免单独滑块受扭矩，同时保证运动的准确性。扭矩方向上由于两个方向的静额定扭矩 Mp 和 My 数据相差不大，安装方向上并没有非常讲究，采用了比较利于空间安排的布置（当初一拍脑门没多想，现在看来应

该采用更利于装配的安装方向）。

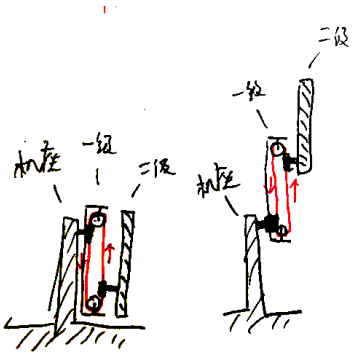


不同的静额定扭矩方向定义示意



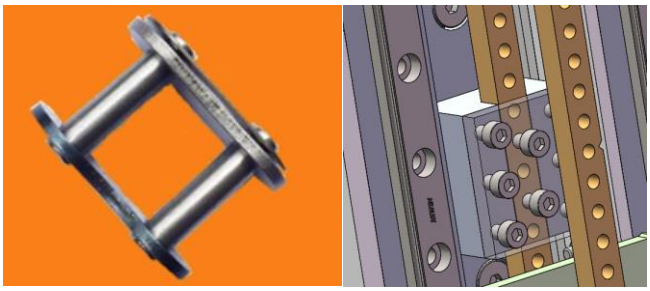
两段滑轨+滑块会同步进行抬升

抬升的动力为一个原减速比的 M3508 电机，我们使用了链传动来将转动转为平动。通过将链条的两侧固定到机座和二级抬升框架上，我们能实现两级抬升的同步运动。

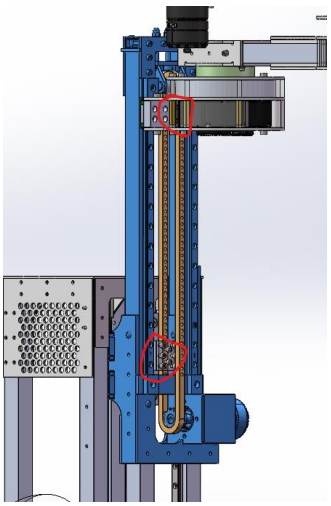


二级同步抬升示意图

链条的型号我们选择 04c，这一块没有进行很仔细的校验，因为是队里祖传的方案，强度应该足够。（根据东大的开源，03c 的强度好像就够了，但 03c 太小了不太好固定，所以还是选择 04c）链条和机座/二级框架的固定采用夹板+螺丝固定。04c 的链条刚好能穿过 M3 的螺丝，这里使用半牙的 M3 螺丝，稍微增强一点抗剪的能力。

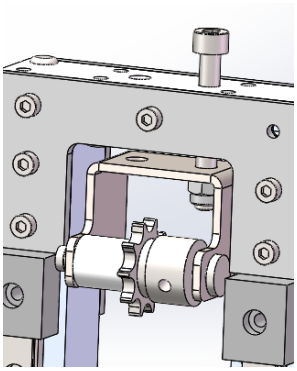


链条的固定



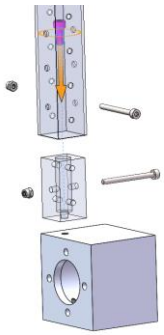
两处链条固定处

一级抬升框架用来固定链轮和电机。在上方链轮处，有可动的张紧结构，此处通过改变链轮中心距来进行张紧。在塞打螺丝与上板间放有压簧，通过调节两根 M6 的螺丝，可以改变中心距实现张紧。这部分制作了一个钣金零件来保证强度，材料为冷锻钢板。



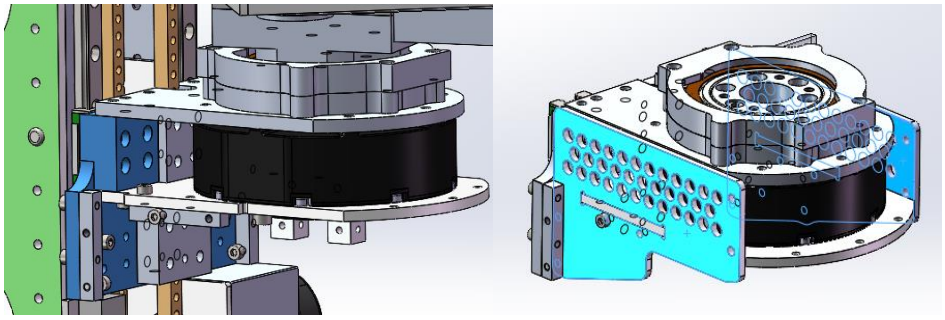
张紧结构

下方的电机固定处，这个地方做了一个电机座的工件。通过上面一个铝方管嵌入件和铝方管连接，这样使电机座和固定滑轨的铝方管稳定的连接。这里的设计和尺寸参考了 2022 年南京理工大学的开源，在此表示感谢。



电机座连接

第一关节（抬升机构）和第二关节的连接处为一个铝工件。上方夹住两块板，用于固定电机，侧面和工件下方通过侧板连在一起，保证关节整体的抗弯强度。

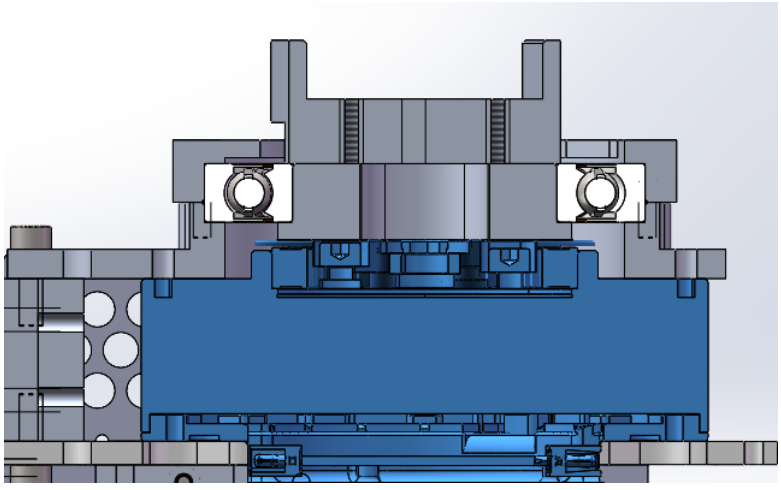


一二关节连接处

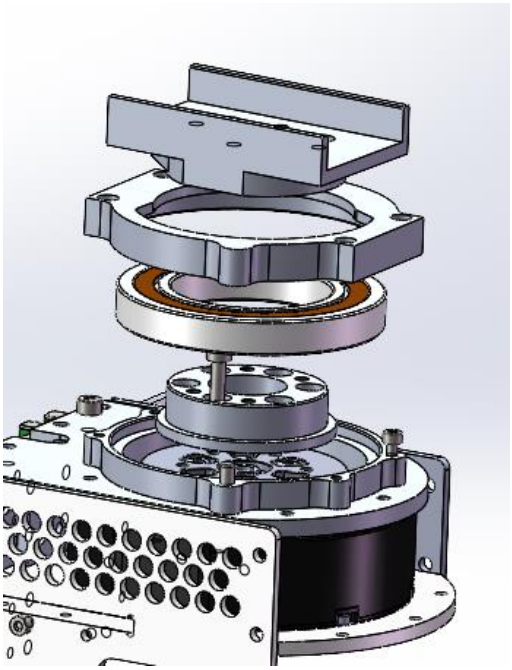
第 2/3 关节

第二/三关节为旋转关节，轴系和连接处设计都相同，在此一并介绍。关节二和三的连杆长度都为 215mm，具体见前期仿真和需求分析。

轴系截面图和爆炸图如下。



轴系截面图



轴系爆炸图

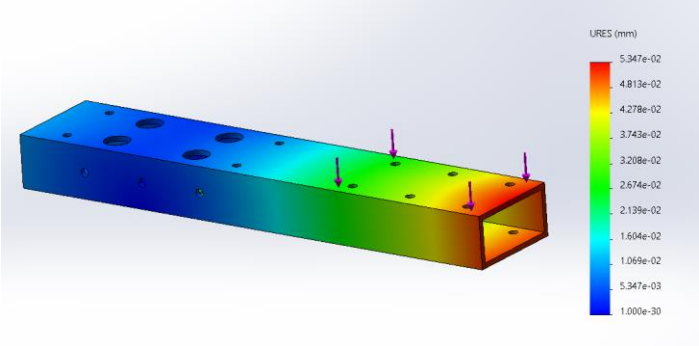
电机采用宇树的 GO-M8010-6 电机，轴承皆采用 16009 深沟球轴承，经校核满足安全需求。此处静止时具有为较大的轴线负载，运动时也具有若干的径向和轴线负载，深沟球轴承在此次本身其实不太适用。但是此处空间比较紧张，角接触球轴承等轴承厚度较大，最后还是向空间妥协，采用了厚度较小的深沟球轴承，但是选用了内径较大的型号，以保证不会失效。

轴承下方是一个铝制工件，用于固定电机和轴承外圈，同时通过控制公差确保装配的同心度。

电机输出为法兰，有 6 个 M4 螺丝孔。考虑到轴承的尺寸和安装的空间，另外设计一铝制工件实现延长输出法兰的作用，与电机输出法兰连接用的 6 个螺丝一旦安装不再拆卸，上面的 4 个螺纹孔用于连接连杆（连接到下一个关节），调试时可多次拆卸。

电机与延长输出轴的工件配合后

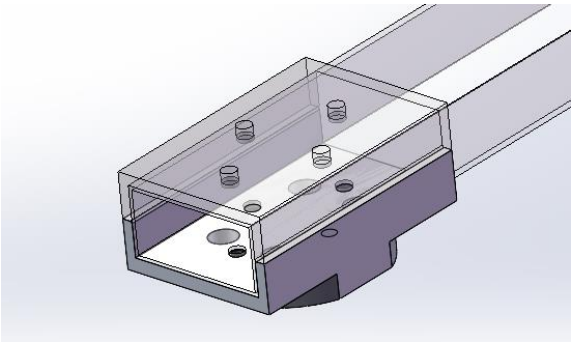
与连杆连接处为用直接螺丝固定到延长法兰的工件。连杆本体为 2040*2 规格的铝方管，经静力学仿真，位移符合要求。



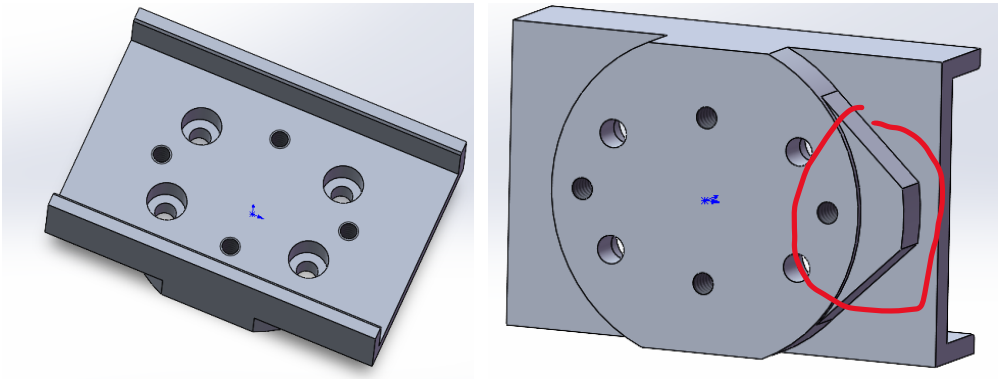
方管静力学仿真

连接处下方为一铝制工件，通过方管的外形和中间孔压紧和方管进行配合，上方的工件是一 U 型的 POM 材料盖子。关节连杆通过四根Φ5 的塞打螺丝与下面的延长输出轴工件连接，方便单独拆卸。

同时，这个工件和轴承的外圈压紧盖配合，还具有限位的作用。



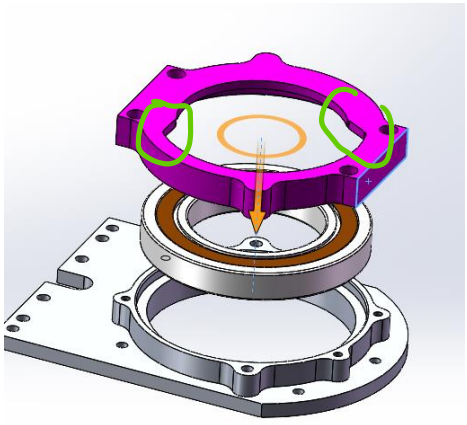
关节连杆连接处



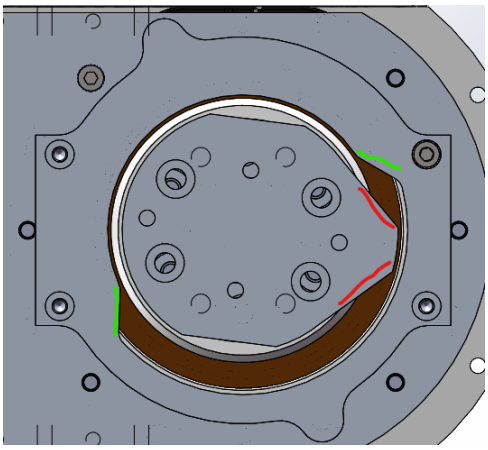
零件的正反面

该零件的下部一侧有具有两个斜面的突起（上图红圈），这里是用来做关节机械限位的。轴承外圈下方跟固定电机的工件配合，上方还有一个压紧的工件，这个工件内部也具有两个斜面（下图绿圈），刚好与里面的关节连杆工件的斜面配合，如下图所示。第 2 和第 3 关节的限位角不同，但原理一样，零件上只有这个轴承外圈上夹紧的零件不同。

这里两个限位的工件都是相同材料的，以避免出现不同材料导致磨损的情况。采用斜面是为了增大接触面积，减少应力的同时也能降低磨损。此轴系的铝制工件都是 6061 铝合金。

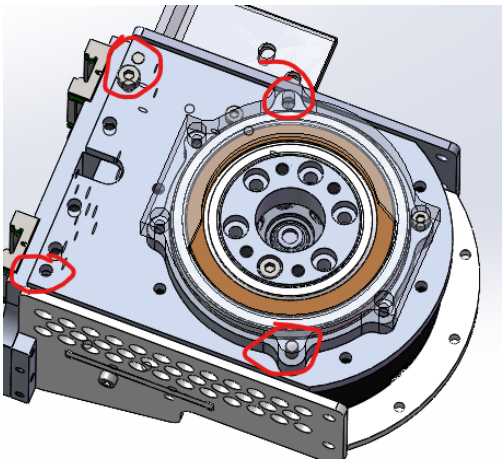


轴承外圈上夹紧盖



限位示意图

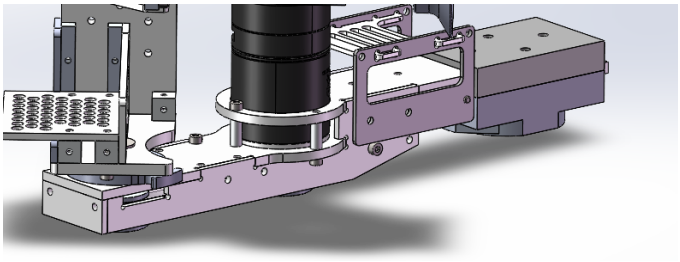
如图所示这 4 个地方放置Φ4 的不锈钢销轴，一方面提高装配的精度，另一方面能够承担一部分剪切的应力。



销轴装配位置

第 4 关节

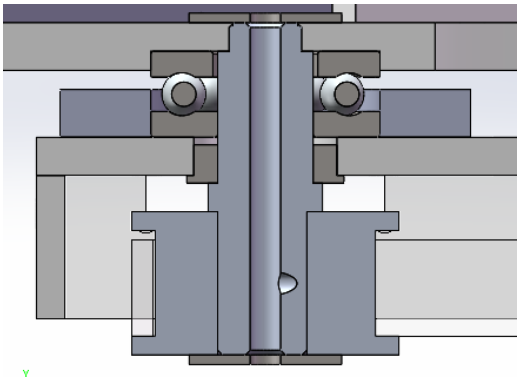
第 3 关节-第 4 关节的连杆部分为 4 块（碳纤维）板材连接而成，壁厚为 3mm，抗弯强度经计算是足够的。



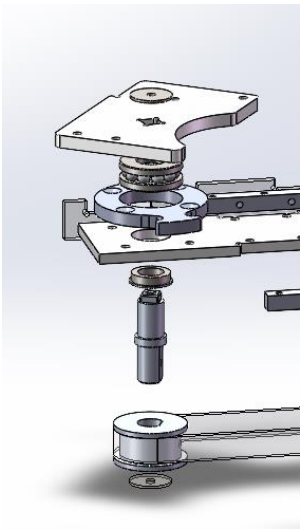
连杆主体为板材

第四关节采用官方减速箱的 M3508 电机，为了空间上与资源岛不干涉，采用了同步带传动，使关节的电机后置，空间上也尽量希望第 4 关节的纵向高度尽量小。同步带选用的是 3M 系列。

第四关节往后的机械臂+矿石的总量约为 4kg，转动惯量本身不太大，但是可能会受到弹丸击打/与墙或者车的撞击等。一旦进入生死局，免不了操作手给你撞个大的，因此外露的传动的地方强度一定要达标。这里使用了一个推力球轴承+一个深沟球轴承，推力球轴承会主要承担静态的总量负载，同时因为同步带选型时有意让它具有一定张紧力，所以加了一个深沟球轴承来承受径向的力。轴系剖面图和爆炸图如下：

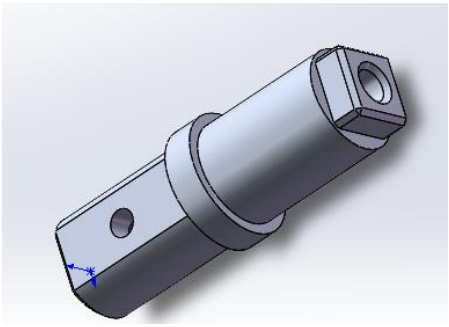


轴系剖面图



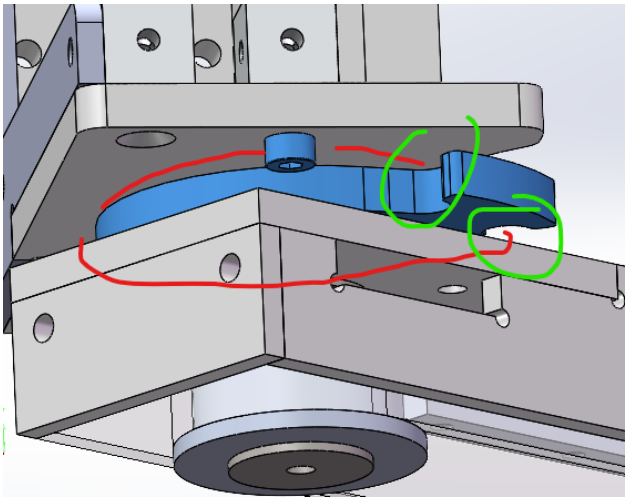
轴系爆炸图

传动轴为一钢（45 钢）制工件，下方为 d 型轴与 d 孔标准件同步轮连接，上方为直径为 7mm 的方轴，与上面的板材（下一关节）连接。轴中间为Φ3.2 的通孔，中间穿过一根 m3 的螺丝，把上面的板材（下一关节）与传动轴压紧。传动轴的零件第一版采用的是铝制，后面方轴的部分担心截面面积过小（后文末端执行器“第七关节”处有类似设计发生过断裂），发生剪切失效，再不更改其他零件的情况下，选择更换了强度更高的材料。



传动轴

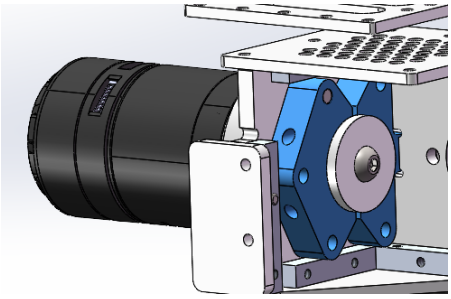
第 4 轴的限位靠的是上方与推力球轴承配合的这个板状零件。上方下一关节的板材下方的某个连接的螺丝头与限位板配合，起到限位的作用。上方的这个螺丝最好使用不带滚花的螺丝头的螺丝，可以适当添加垫片调整配合的深度。



4 轴限位示意图

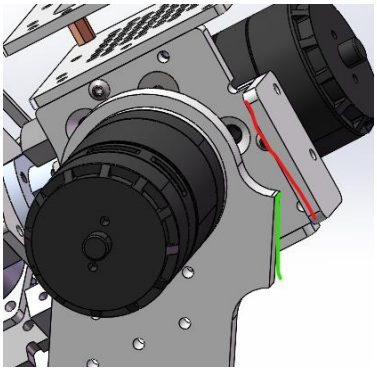
第 5 关节

第五关节是改过减速箱减速比（改为 1：27）的 M3508 电机直驱，第一代使用的是原装的 1：19 减速箱，后面分区赛发现连打 3 小局的话会很明显的发热，于是改变了减速比。结构设计平平无奇，使用了经典款上交联轴器。



上交同款联轴器

限位是图示两个平面，这里只做了上限位，第五关节水平时即到上限位的地方。这里机械上没有下限位，电控有做下方的电限位。

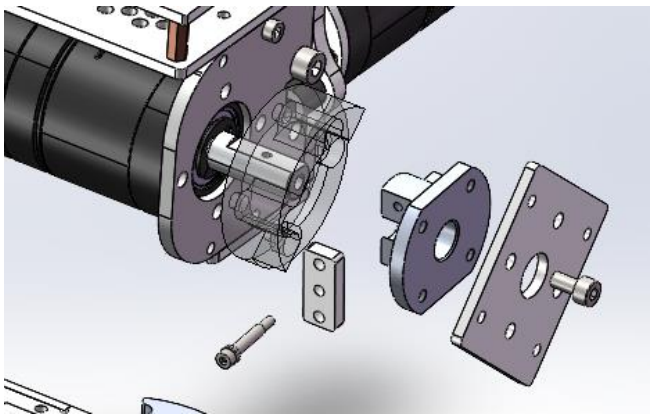


5 轴限位示意图

第 6 关节

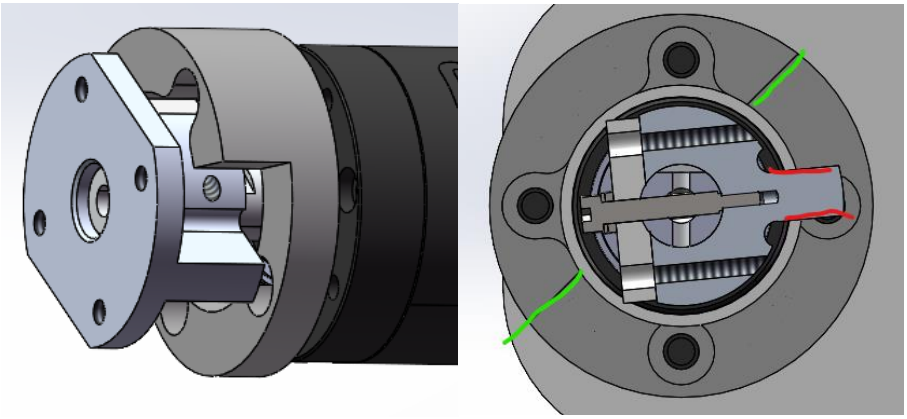
第六关节是官方减速箱的 M3508 电机直驱，为了减少到末端（吸盘末端）的长度，需要尽量压缩电机输出轴连轴的空间。这里设计了一个连轴的法兰工件，一方面跟下一关节连轴，另一方面直接把限位也整合进去。

这里连轴的原理同上交联轴器，通过两个零件来夹紧电机的 D 型输出轴。同时，这里把输出轴的 M3 螺纹孔打穿成大于Φ3 的通孔，中间插入一根Φ3 的塞打螺丝，同时起到轴向固定的作用。



6 轴爆炸示意图

限位是通过连轴工件的一个突起和电机定子处的一个 POM 材料工件的形位配合，如下图所示。



6 轴限位示意

1.5.1.3 末端执行器

需求分析

我们把末端执行器视为一个整体，从机械臂第五关节的转轴到末端执行器最末端的距离，即为第六关节的连杆长度。为了减少兑换时 pitch 角变化引发的 z 轴高度的变化，我们会希望第六关节的连杆长度越短越好，也就是说，希望末端执行器这部分的总长度越短越好。

当然，从电机负载的角度来看，我们也希望这部分越轻越好。

因为机械臂末端三轴是正交的三轴，会发生 rpy 的旋转变化，我们希望末端执行器整体的空间越小越好，为了尽量避免在三轴运动过程中和机械臂本体的空间干涉。

总结末端执行器的需求如下：

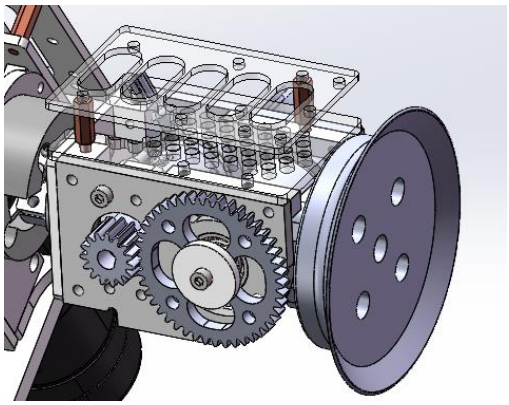
- 基本的结构刚度
- 距离第五关节的轴线的长度越短越好
- 整体质量越轻越好
- 整体空间越小越好
- 便于电气布置，便于维修

除了以上基本的需求以外，在不断迭代测试的过程中，我们发现了一些额外的问题。我们的机械臂整体横向长度过长，伸至最长时刚度是最低的，但是我们取矿时即需要把机械臂伸到最长。我们原本的计划是从小资源岛的侧面取矿，但是实际测试发现由于机械臂的刚度不够，从侧面取矿会导致矿石倾斜，以至于在资源岛的凹槽中发生自锁，导致矿石完全卡住，取矿失败；同时从分区赛的录像看来，机械臂方案从侧面取矿的，整体成功率其实都不高，纯机械臂方案本身刚度就较低，如果操作手操作失误更会雪上加霜。

综合来看，机械臂的刚度提升的改进是有限的，我们希望从原理上杜绝导致取矿失败的因素。

我们希望给末端执行器增加一个旋转自由度，使吸盘能够垂直向下，这样我们可以从正上方获取矿石。经测试，从正上方取矿的成功率可以达到 100%，且容错率极高。

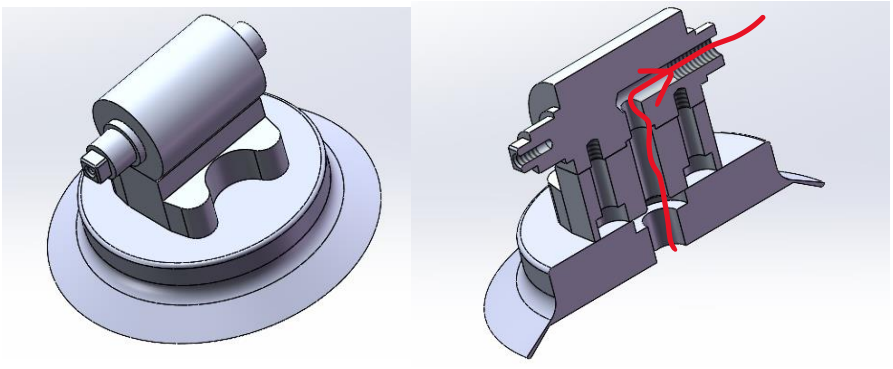
结构设计



末端执行器整体结构

末端执行器整体由末端吸盘及气路元件和驱动的电机及传动结构构成。

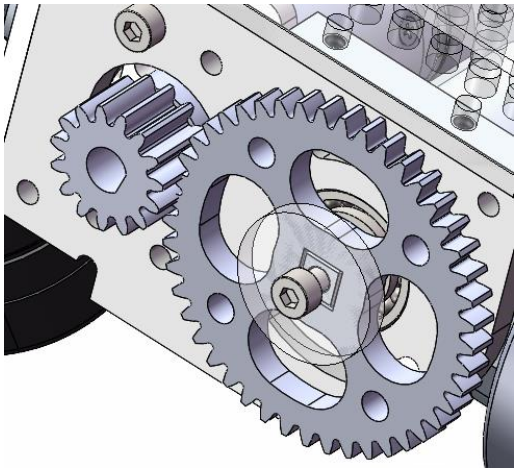
我们的吸盘选型为 H80 黑色丁腈橡胶吸盘，选用单层且不带海绵的吸盘，因为银矿的姿态非常确定且端正，单层吸盘不需要任何特殊处理即可从上方稳定的获取银矿。吸盘背后的进气金具，我们没有采用原厂的官方金具，因为体积过大，难以减小；这里我们自己设计了两个零件，一个轴件，一个和吸盘连接的法兰件，这样使进气的路径过吸盘旋转的轴心，外面接一个旋转快拆接头，做到空间最小化的同时能够优雅的走气。



吸盘和自指进气件（红色为气路）

驱动部分，我们使用了原装的 M2006 电机，它的最大扭矩为 1NM，参考前面我们说的第五关节原装 M3508 电机在打完 3 小局后会有发热严重的现象，原装 M3508 电机的最大扭矩为 3NM，所以这里我们加了一级齿轮传动，减速比为 1：3。这里为了空间尽量小，用了 1 模的齿轮（小齿轮是我能在广发传动店里找到的最小的齿轮，大齿轮的齿数根据它和减速比来定的）。大齿轮是线切割加工的钢质齿轮，

中间的传动孔是一个 6mm 的方孔。这里为了轴向固定在轴工件上加了一个 M3 的螺纹孔，但是在测试中被撞击后方轴被剪断，鉴定为此处截面面积过小，方轴的轴径应该变得更大。



齿轮传动示意图

1.5.1.4 4 矿石储矿机构

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：

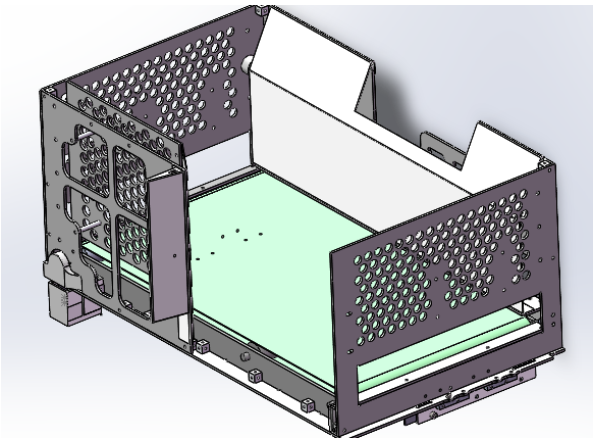
1. 能储 2 个矿，加上吸盘自己带一个可以总共完成 3 个矿的同时储存与运输；

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 200×200×200 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；

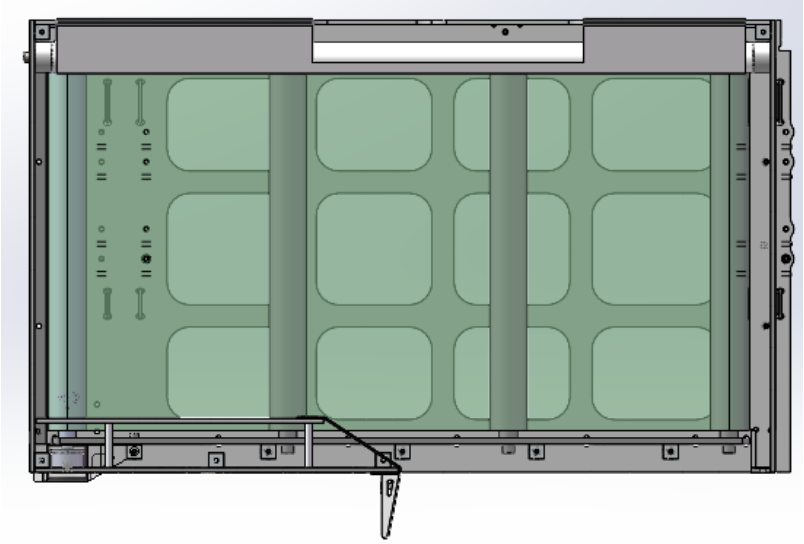
结构设计



储矿机构

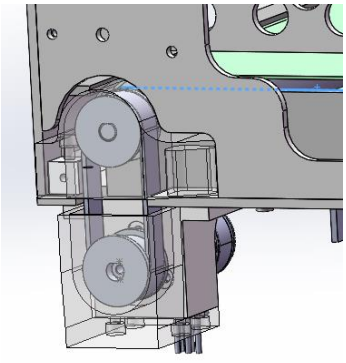
本储矿机构主体为四周的挡板和下方的皮带机构组成。

下方的皮带机构由滚筒和皮带组成，动力来源为一侧的 M2006 电机带动一根主动滚筒。滚筒一共有 4 根，一根为主动，3 根为从动。左右两端的中心距为 450mm，比两个矿石稍大；两根滚筒间的间距为 150mm，间隔距离为为比一个矿石稍小。主动滚筒为定制工件，输出轴为 d 型轴，为了配合使用 d 孔的标准件同步轮，表面有滚花；从动滚筒为标准件，选用了能找到的负载较小的型号（WT-01），里面自带轴承，只需要简单固定两端的轴即可使用，非常方便。



滚筒示意图（左侧为主动）

主动侧为同步带传动，电机置于下方。选用 3M 系列同步带。且设计有便于拆卸的防尘盖设计，防止传动因为异物失效。



主动侧传动和防尘盖

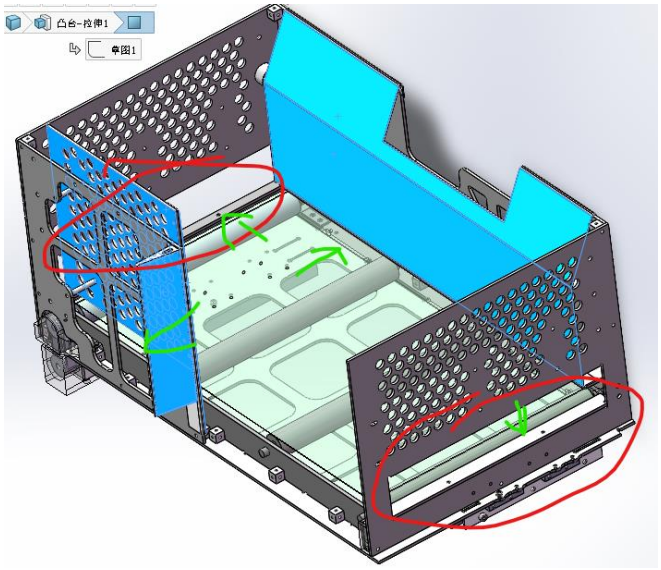
整个储矿机构的矿石可进入空间大小为 230*494*xx（高度可延申，因为上面是开口的），实测这个尺寸配合我们的机械臂的位置精度，容错率非常高，无干扰取矿+储矿的成功率可达 100%。允许的矿石歪斜

角度也很大，两侧歪斜 45° 的情况都可以成功储存进去不卡。

四周的结构挡板全是 3mm 的（碳纤维）板材，坚固耐打。

储矿空间内/外的一些地方做了斜面，主要通过弯折 PVC 板或者 3d 打印来实现（下图蓝色）。

两侧有宽 43mm 的槽（下图红圈），用于在内部进子弹的情况下能够把子弹挤出去。除此之外，所有的缝隙（下图绿色箭头所指）全部设计得小于 17mm 子弹的半径，反之子弹卡进去造成传动失效，一些难以结构封死的地方后期全部用铁氟龙胶带挡住同理。



一些储矿机构细节示意

1.5.1.5 底盘

需求分析

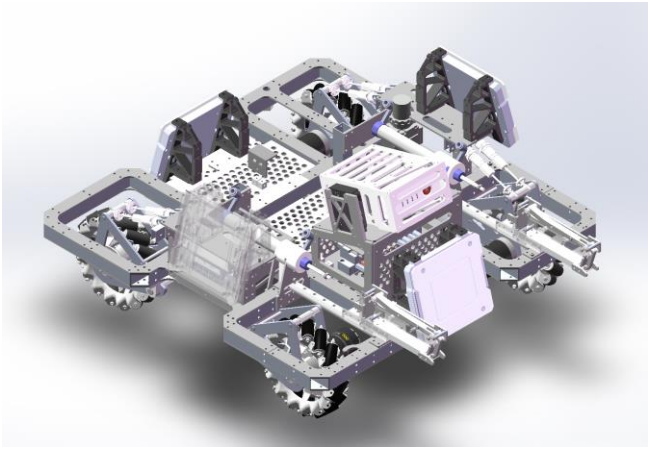
底盘是机器人上层机构的机座，同时机器人提供移动功能。在 23 赛季的规则中，起伏路段大幅减少，对于我们只取小矿石的取矿-兑换的路线来说，地面全是平整的地面，因此对底盘的减震性能要求并没有那么高。同时，在对矿时最好能有全向移动的功能。除此之外，上层机构（机械臂）运动时，会对底盘产生反作用力，可能造成底盘打滑位移，因此底盘还需要具有对上层机构运动的阻尼减震性能，最好还能相对地面完全锁定。经过分区赛的检验，发现初版底盘虽然满足需求，但在高强度的对抗比赛中整车强度欠佳，在分区赛后期底盘出现了部分板材断裂，关键处铝方管形变等问题。考虑到国赛极可能出现复活赛与正赛连打的情况，工程车整体需要加强强度，以应对连续、高强度的比赛。

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）
4. 强度高，能经受高强度比赛。

结构设计



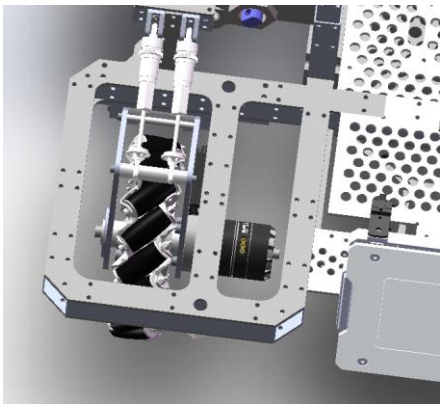
底盘图纸

底盘参数：

名称	*参数*
尺寸（长*宽*高） *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*	*590*590*124（mm^3）*
下框架离地高度	*100mm*
重量	*12.5kg*
接近角	*53.7°*
通过角	*53.6°*
轮距（横*竖）	*438*380（mm*mm）*

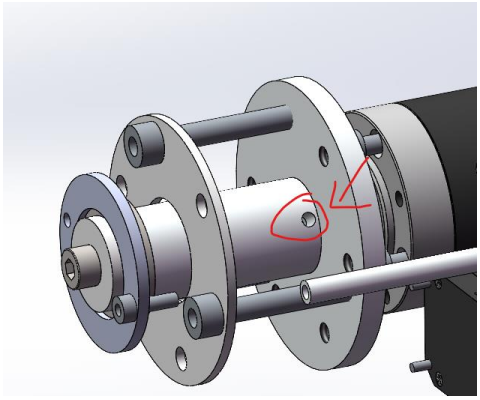
底盘框架主体为 2020 规格的铝方管和 2mm 的玻纤板相铆接，通过玻纤板形状对齐来保证装配精度。同时这样做会增加底盘质量，工程车的上层机构质量较大，因此底盘质量变高有利于重心的降低，防止翻

车。同时玻纤板还会在碰撞时承受冲击，通过玻纤板间挤压减小铝方管所承受冲击，有效减小碰撞时底盘铝方管的形变。



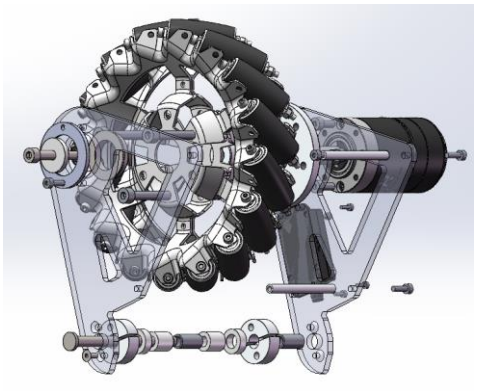
板材对齐保证装配精度

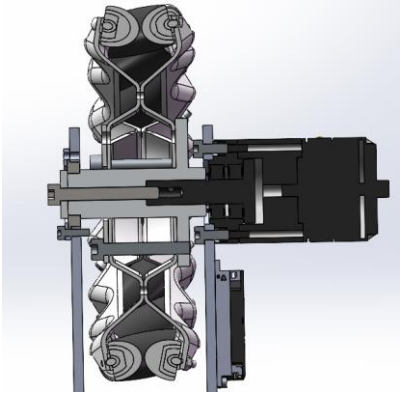
轮组部分，通过一个机加件将电机和麦轮连在一起。机加件和麦轮通过一根Φ3 的销轴进行周向的固定。



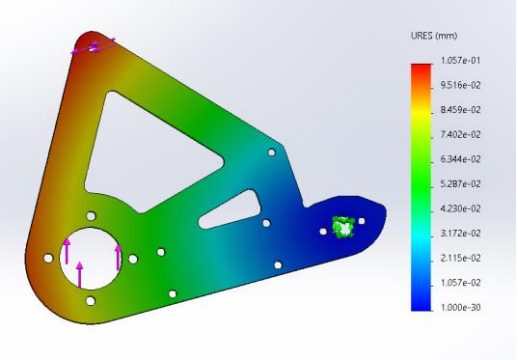
销轴位置

轮组为纵臂式夹轮结构，两块板固定在麦轮的两侧，防止车轮出现“内外八”的情况。轮组的装配顺序为从左往右依次装配。





轮组装配顺序和截面图

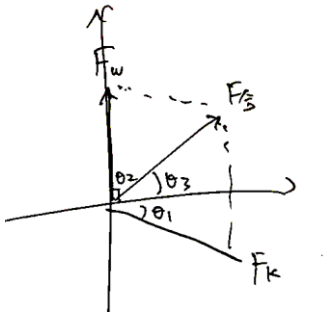


轮组侧板仿真结果

轮组两侧的夹板为两块厚度为 4mm 的玻纤板，经仿真最大位移为 0.1mm，符合要求。

悬架为纵臂式避震。轮组的悬架转轴处为Φ6 的卡簧销轴，经校验能够承受 35kg 的工程车在赛场上的各种急停、冲撞的工况。

一共有 8 组避震器，选用的避震器最大行程为 30mm，以静止状态弹簧压缩量为 13mm 计算。



悬挂受力分析

其中，θ_1=-15.83°，θ_2=90°，θ_3=40.87°。

$$\tan\theta_3 = \frac{F_k \sin\theta_1 + F_W \sin\theta_2}{F_k \cos\theta_1 + F_W \cos\theta_2}$$

$$F_k = \frac{F_W(\tan\theta_3 \cos\theta_2 - \sin\theta_2)}{-\tan\theta_3 \cos\theta_1 + \sin\theta_1}$$

求得 F_k=65.69N.则单个避震器 F_k=65.69/2=32.85N.

则弹簧弹性系数 K=2.53e3(N/m).

弹簧常数公式（单位：kgf/mm）：k=(G*d^4)/(8*Dm^3*Nc)

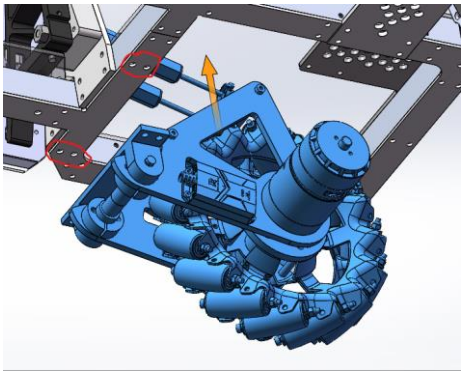
$$k=(G\times d^4)/(8\times Dm^3\times Nc)$$

G=线材的钢性模数：琴钢丝G=8000；不锈钢丝G=7300，磷青铜线G=4500，黄铜线G=3500

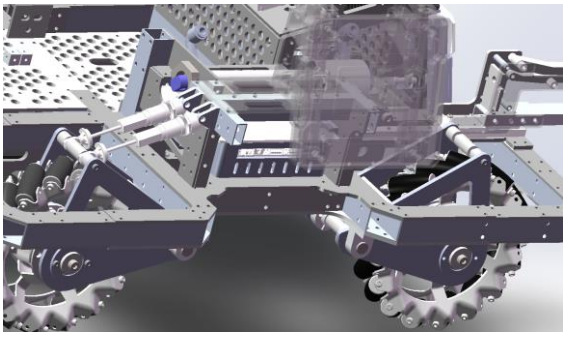
根据弹簧线径公式，选用不锈钢丝弹簧，代入相应尺寸数据，求得避震器弹簧线径 d=2.1mm。所以**选用线径为 2mm 的不锈钢弹簧**。

轮组和底盘的连接通过 4 个 M4 的螺丝连接，较为模块化。在场上一旦出现问题，能够整个轮组模块迅速的整体替换下来。

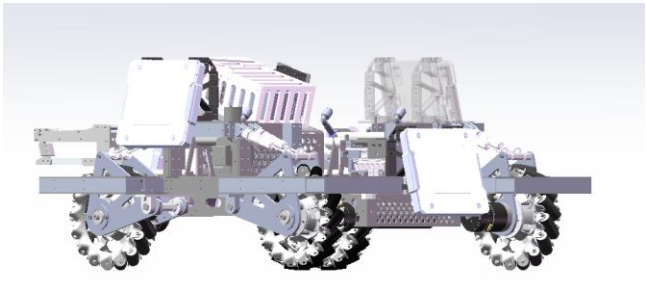
轮组和底盘的连接



这版底盘选择将避震器另一端与铝方管相连接，机器人在地面运行时避震器只会受压而不会受拉，故在前后轮的避震器间选择以一根 1020 铝方管连接，这样可以有效利用连接处与底盘间的空间。通过这种方法，可以放入减压阀与 NUC，有效节省了底盘上部空间，有利于其他机构安装与检查。装甲板则选择放置在避震器中间上方，不仅可以有效利用空间，也让工程车在陀螺时装甲板高度有变化，尽量提高工程车受击时的生存时间。

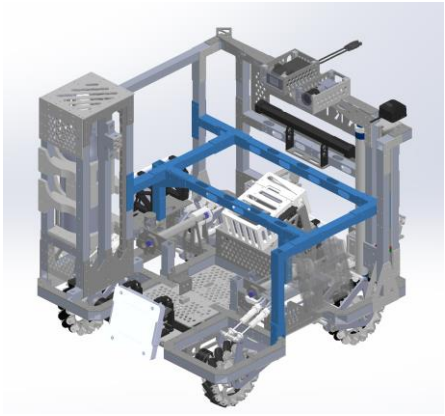


轮组和底盘的连接



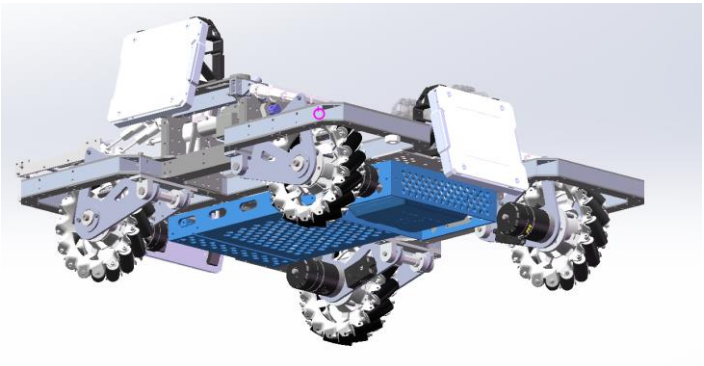
装甲板高度

为加强整车强度，选择在内部增加一个由铝方管搭建而成的结构，用于将工程车各个模块紧密相连。分区赛时为了便于维修工程车，将整车模块化，将连接简单化。但比赛中发现极端模块化也会降低整车强度。故选择放弃分离便利，选择将各部分紧固连接，让整体承受冲击。



工程车内部框架

底盘底部安装了由玻纤板搭建的保护，用于保护气路原件。工程车也延续了上一版的“上电下气”方式，底盘之上布线，下方则仅有气路。这样将电路、气路分离，可减小维护难度，提高效率。



底部保护

1.5.1.6 救援机构

需求分析

基本功能

为了更快更高效的复活己方机器人，需要工程车通过救援机构将战亡机器人拖回基地的补血点，不仅节约复活所消耗的金币，也节省了复活机器人需要回到己方基地激活发射机构所需的时间。为工程车设计救援爪，可以节约对战中非必要的资源消耗，加速己方回场。

需求分析

通过对救援机构设计目的的分析，可知救援机构需要满足的性能：

- 必须满足

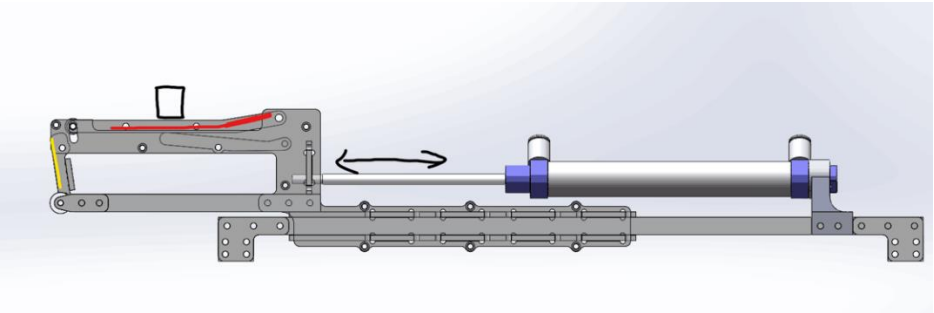
1. 能够抓/钩住战亡机器人，并且可以释放。
2. 在一场比赛内，机构可多次使用。
3. 性能发挥稳定。
4. 在拖拽过程中会经过爬坡、下坡以及不平坦路面，两车不能分离。
5. 结构稳固牢靠，维护简单。

- 最好满足

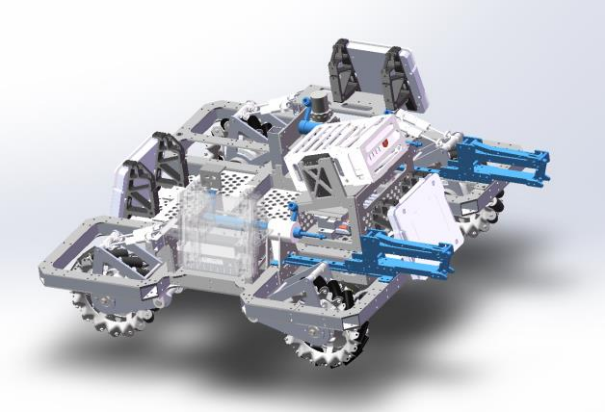
1. 抓/钩取容易、分离方法简单。
2. 操作手操作简单，不需要精细的位置调整。（允许操作误差）
3. 寿命长，拆卸、更换简单。
4. 占据小空间，不影响工程车与被救援机器人其他机构的设计、摆放。
5. 动力源需求少。

因此版工程车上方空间大部分已被占据，安装竖向的旋转式救援爪很困难，且大概率会与装甲板检测区相干涉，选择使用往年救援爪方案。

结构设计

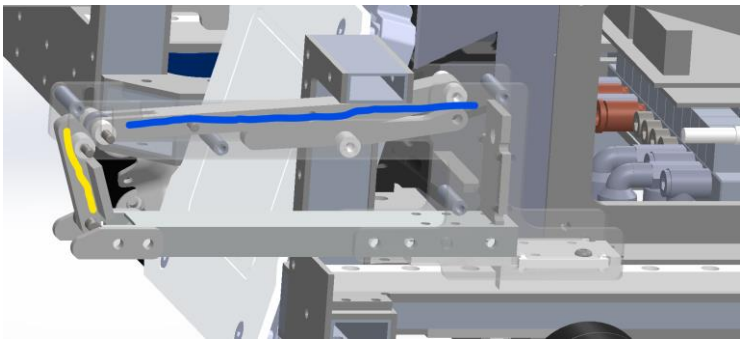


救援机构结构



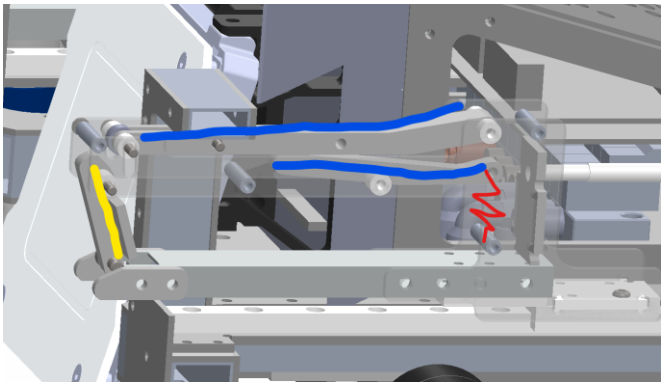
救援机构安装位置

通过气缸为动力源，推动机构在水平方向运动。在气缸伸出时，连杆与上方铝方管相互挤压摩擦，使黄色部分形成单向门。在气缸收回时，连杆归位，使黄色部分可以自由旋转，完成释放。通过此种方式，有简单、高效的优势。



气缸伸出时救援爪机构状态

当气缸伸出时，上方铝方管会挤压蓝色部分连杆，将其前端下压。此时蓝色杆与黄色部分形成单向门结构，黄色部分只能向内旋转而不能向外打开。此状态下可以通过用救援机构直接撞击被救援车辆的防护架完成对目标车辆的抓取，且因为单向门结构在工程车运动期间两车不会分离。



气缸回收时救援爪机构状态

当气缸回收时，蓝色连杆部分不再受上方铝方管挤压，通过红色处拉簧拉动连杆，将连杆前端抬起，连杆不再阻碍黄色部分运动，此时单向门结构被解除，单向门失效，黄色部分可以自由旋转，机构内部被抓取的防护架可以自由向外离开。通过收回救援机构，可以直接释放被抓取车辆。

虽然救援爪以安装在上场工程车之上，但因设计者测试不及时，在临赛时才发现此机构有着重大问题：在单向门结构受力向外拉时回收救援爪会导致单向门与连杆机构卡死，导致机构失效，无法释放。考虑到临近比赛，且更改相关结构后没有解决此问题，工程车救援功能依旧有缺陷，故放弃使用此机构，上场时放弃工程车救援。

1.5.1.7 图传抬升机构

需求分析

图传模块为操作手提供视野，图传模块安装位置好坏直接决定了操作手的视野情况，影响着工程车兑换效率与移动可操控性。因工程车操作手需要在小资源岛取矿，在兑换站兑矿，在战术上需要精准把控位置，阻挡对手车辆。视觉识别兑换站姿态也需要相机有良好视野，故工程车的图传、相机有以下要求：

1. 视野良好，遮挡少。
2. 能看清整车状态与位置。
3. 最好与兑换站中心高度齐平。
4. 图传与相机受保护，不被弹丸击打。
5. 便于安装、维修。

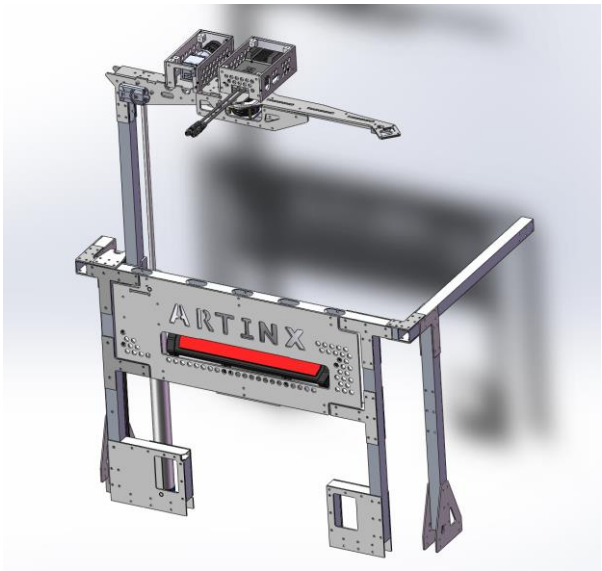
在往届与其他队伍方案中，龙门架方案的工程车倾向于将图传模块安装于车辆前侧中间，紧贴底盘，

视角上仰。但此位置不利于操作手操控车辆，且可视范围有限。机械臂方案工程车有将图传模块安装于机械臂末端连杆，但视野受机械臂抖动影响。也有额外安装摄像头，在内部“看监控”。但此方案图像帧率与分辨率均有限，不予考虑。

最终选择搭建一气动抬升平台，在其上放置图传、相机。此方案可为操作手提供极佳视野，但不可避免在底盘移动时图传会晃动，在平缓路段时晃动在可接受范围内，故使用此方法。

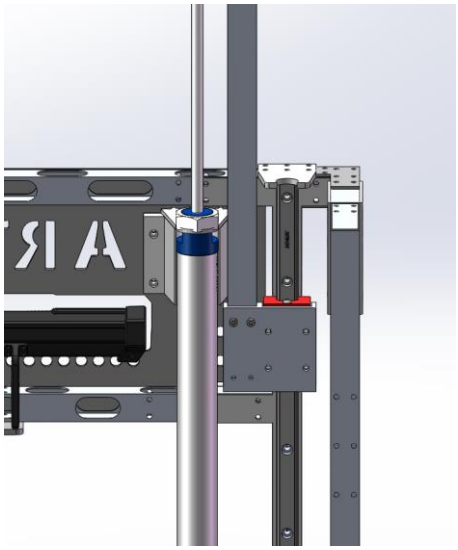
结构设计

图传抬升机构总览

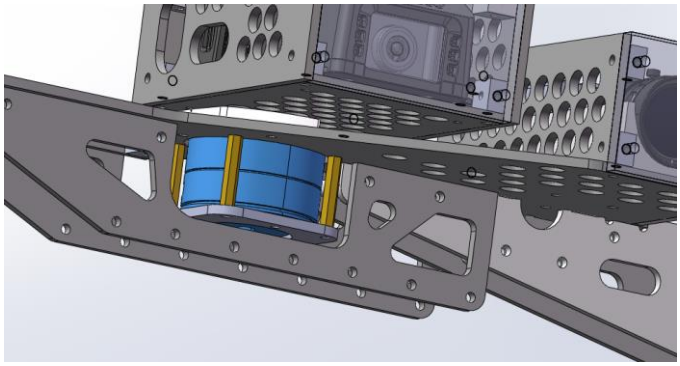


由于是一次性抬升机构，这里我们在气路上加上了单向阀和手动的泄气阀，来保证赛场上即使漏气也不会导致图传错误的下降，只有在下场后我们会手动进行泄气下降。

机构选择气缸与滑轨滑块结构，同时为图传提供水平方向旋转的自由度，用于在非兑换时将图传旋转 180 度，使工程车朝向后侧驾驶，优先受打击后侧，减小机械臂受打击可能性，保护机械臂。



抬升机构

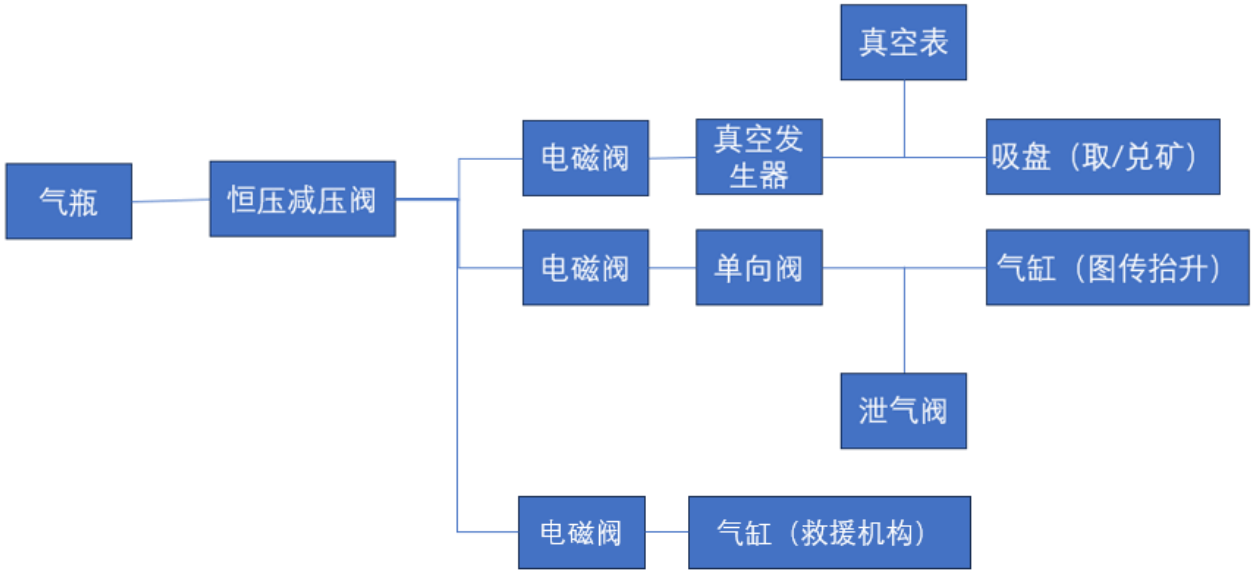


图传下方电机

因临赛前取消救援爪机构，同时工程车在大功率运行时接地或与其他车辆碰撞时会有大电流流经工程车，导致阀岛电磁阀错误关闭。经组长考虑，以后侧为正向开车的方式意义不大，故取消了图传自由度。

1.5.1.8 气路系统设计

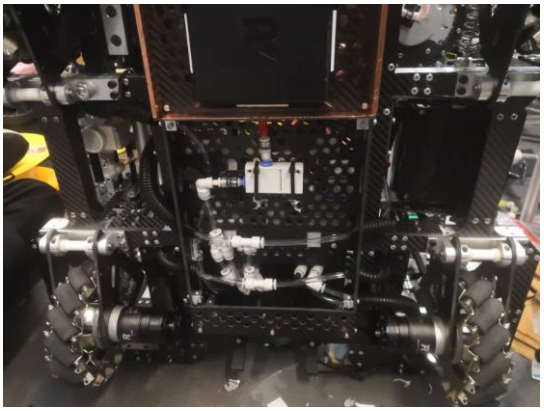
我们的工程车有许多气动元件，气路系统示意图如下：



气路系统示意图

在结构的设计上，我们采用了集成的电磁阀阀岛方案，阀岛位于底盘中央后方的位置，往底盘下方出气管，再走到各个执行器，底盘上方是电路元件，底盘下方是气管，实现电和气的分离。

集成阀岛方案



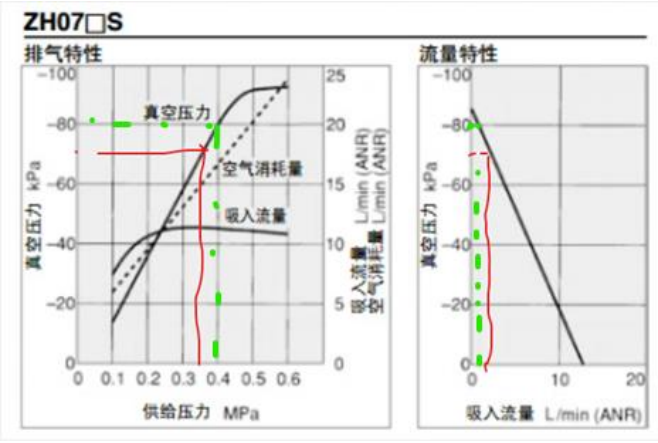
底盘下方是走气的空间，方便维护

一些关键元件的具体参数如下表：

气瓶	容量：0.81L 上场储存气压：20Mpa
减压阀	型号：AR20-02E-B 上车输出压：0.45Mpa（仅供参考，与实际电磁阀的流量有关）
真空发生器	型号：zh07bs 上车工作真空度：-80kpa
吸盘	型号：zptH80

我们的真空发生元件采用的是真空发生器，型号如上表。经由理论计算，真空度达到-22Kpa 即可拿起矿石，但是在测试过程中发现矿石表面的磨损程度和表面缺陷会极大的影响拿起矿石的动作，在-45Kpa 左右，能在不受干扰的情况下拿起损坏程度高的矿石。在实际比赛中，因为会出现各种撞击，在分区赛阶段我们实际上场的真空度是-60Kpa，不受外部干扰的情况下较为稳定。分区赛后我们与其他队伍进行了交流，在国赛阶段，我们又把上场的真空度调高到了-80Kpa，这个真空度下我们打了 29 小局的比赛，没有在操作手误操关气以外的情况以外丢失过矿石。

我们采用的是箱式的真空发生器，型号为 zh07bs，这款真空发生器的特性曲线如下图：



真空发生器特性曲线

在我们上场的真空度（-80Kpa），理论上供给压力应该为 0.4Mpa，但是实际上车的压力实测比 0.4Mpa

更高，约为 0.45-0.5Mpa，应该是因为集成电磁阀阀岛的气流量较小（实际上我们在车上使用了两路电磁阀出气来供给真空发生器，一路确实气流量太小了），加上我们从真空发生器到机械臂末端的气路过长导致（差不多 2m 左右，因为有拖链和弹簧气管）。实际气路在车上布置完成后，我们经过了测试，在保证末端的真空度到达-80Kpa 的情况下，来决定我们减压阀输出的正压是多少，实测我们车上大概是 0.45Mpa（详见 1.6.1.2 测试记录）。

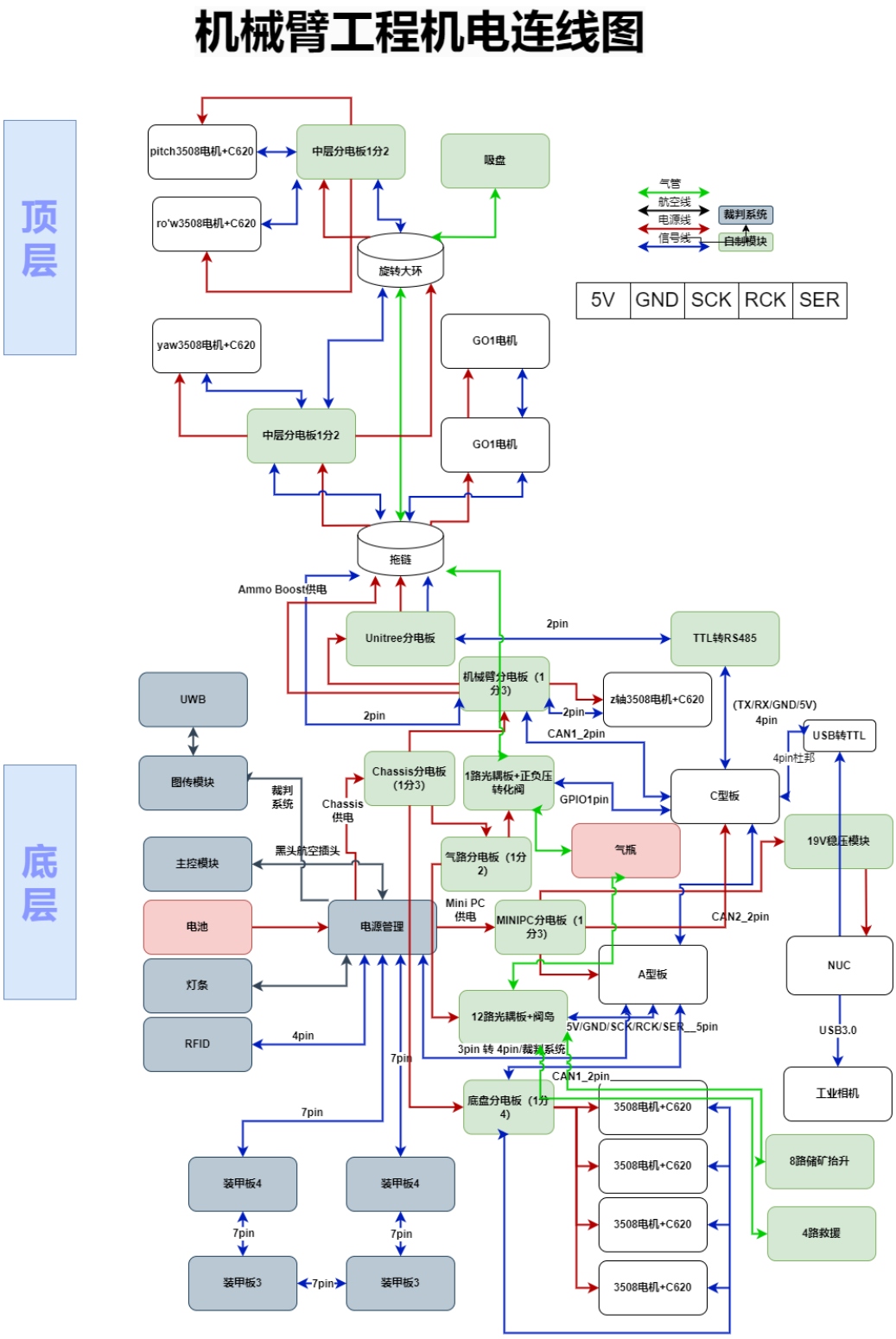
我们在末端真空气路加了一个真空表，位于底盘上，极大的方便了场上的检修。一旦在场上发现末端真空压不够，可以看真空表的示数手动调整供给压力，可以实现场间 3min 的及时调整。在调试时也可以防止出现突然拿不起矿的玄学问题出现，及时找到问题。

机械式真空表

1.5.2 硬件设计

整体机电连线图

工程车上的机电连线图如下图所示：

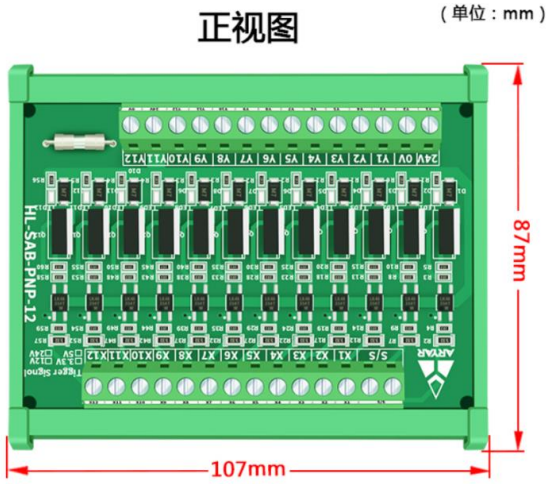


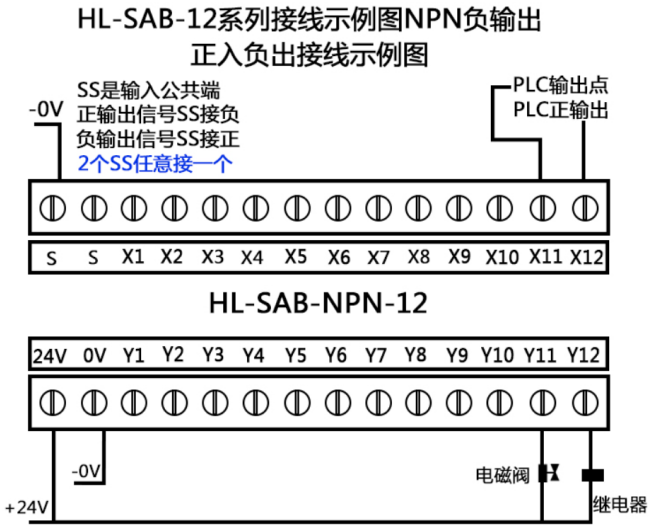
气路自研控制板

传统气驱动是由电磁阀控制气路内部的压强 为正压，负压还是大气压。使用由气路实现的机构，一个自由度 通常需要两路气路甚至更多。考虑到我们延展机构所需要的气路数目不低，那么如何控制多路气

路就成了一个设计点，为了解决冗余的线路问题，我们采取了多路光耦继电器+位移寄存器的控制方式。

光耦继电器

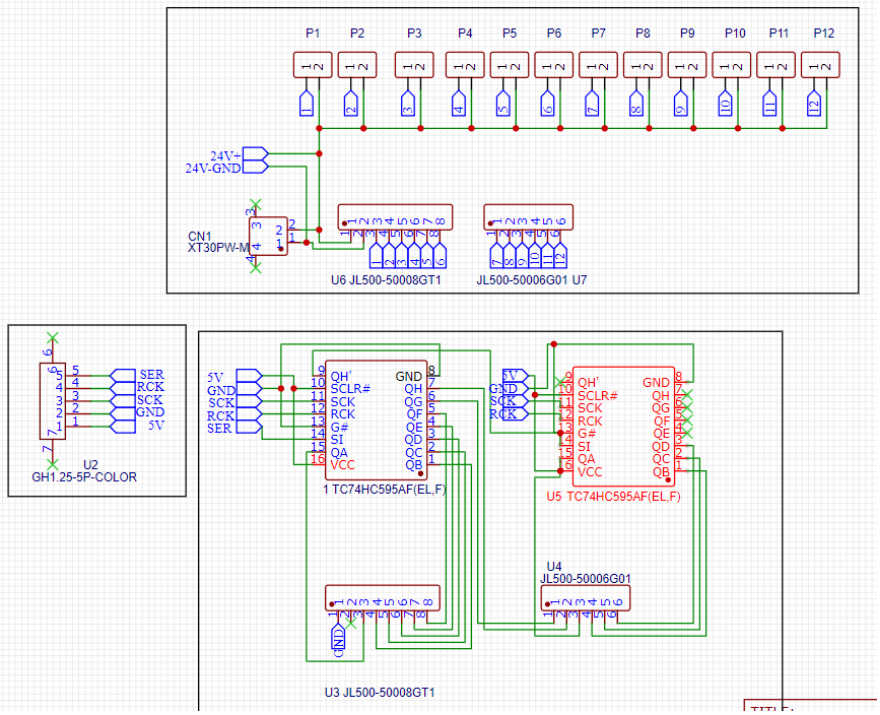


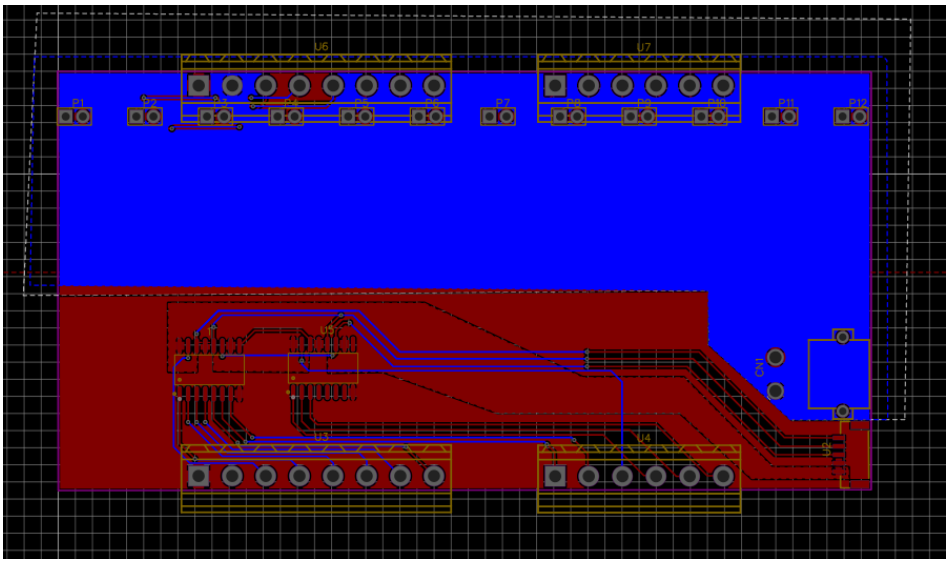


光耦板采用 HL-SAB-NPN-12 和 HL-SAB-NPN-1 型号，其设计均为共阳极的负极输出，方便与气阀连接进行控制。其核心目的在于处理驱动问题，使得控制电路和驱动电路隔离开来，每个 X 输入端对应相同数字的 Y 输出端，当位移寄存器输入高电平，光耦板输出端将输出-24V 电压，在位移寄存器所在电路板上形成 24V 的输出使气路电磁阀导通。

位移寄存器

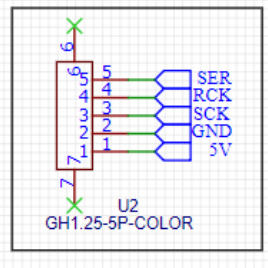
位移寄存器控制板的原理图和 PCB 设计图如下图所示：





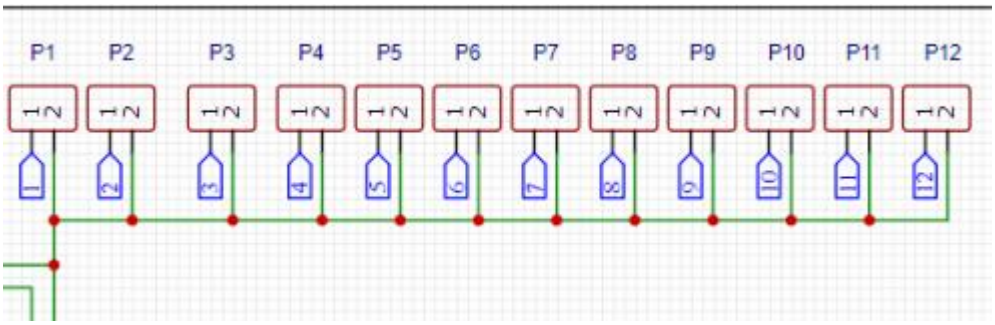
我们可以将其抽象成几个部分：

1. 输入部分：



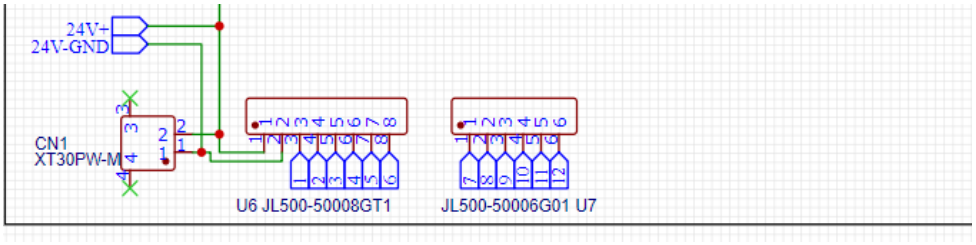
此处为 5pin 卧贴，每一路连接位移寄存器对应的端口。

2. 输出部分：

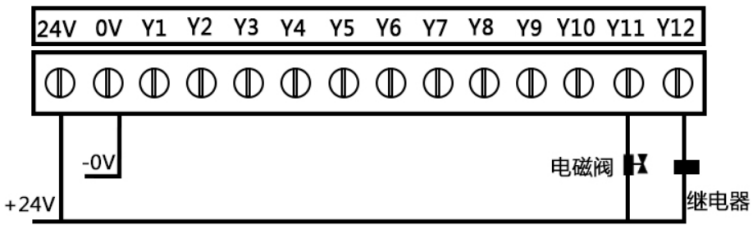


上图 P1 至 P12 分别连接至气阀上，1 至 12 端口为负输出口，另一端为共阳极。

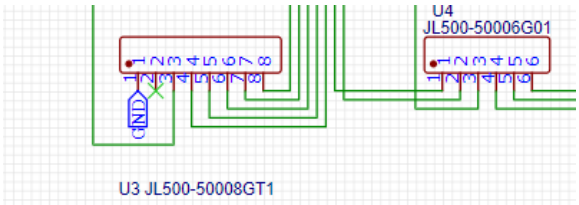
3. 光耦板输出：



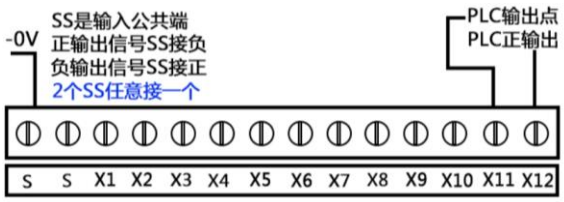
CN1XT30PW-M 为 XT30 卧贴，提供 24V 电源。U6 JL500-50008GT 和 JL500-50006G01 U7 为电路板上焊接光耦板输出口的部分，对应光耦板接线示例图如下部分：



4. 光耦板输入：

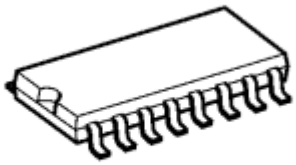


对应光耦板的输入端的焊接部分，对应光耦板接线示例图如下部分：

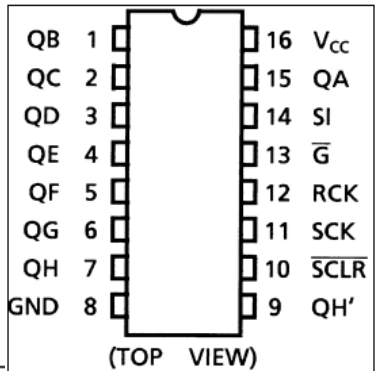


5.寄存器部分：

为减少接线及方便控制，采用位移寄存器进行控制，位移寄存器型号采用 TC74HC595AF(EL,F)。其中 Vcc 供电，GND 接地，SI、RCK、SCK、RCLR 为四个输入口，其他为寄存口的输出。



SOP16-P-300-1.27A



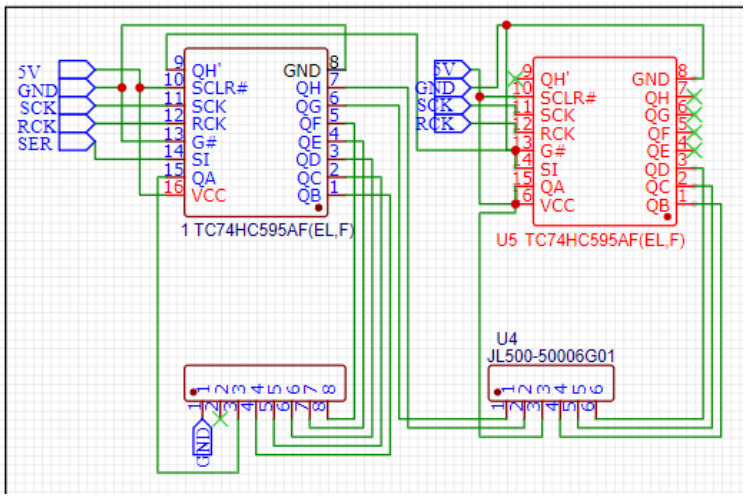
位移寄存器控制逻辑真值表如下：

Inputs					Function
SI	SCK	$\overline{SCLR}$	RCK	$\overline{G}$	
X	X	X	X	H	QA thru QH outputs disable
X	X	X	X	L	QA thru QH outputs enable
X	X	L	X	X	Shift register is cleared.
L	↑	H	X	X	First stage of S.R. becomes "L". Other stages store the data of previous stage, respectively.
H	↑	H	X	X	First stage of S.R. becomes "H". Other stages store the data of previous stage, respectively.
X	↓	H	X	X	State of S.R. is not changed.
X	X	X	↑	X	S.R. data is stored into storage register.
X	X	X	↓	X	Storage register stage is not changed.

X: Don't care

每次 SCK 上拉都将 QA 至 QH 每个寄存口的数据（0 或 1）转移到下一个寄存口，并将 SI 的数据寄存于 QA 口，SCK 下拉时寄存口数据不变。RCK 每次上拉，各个寄存口修改一次输出，每次下拉寄存口输出数据不变。其中 QH’原理如下系统图，QH’连接于 QH 的 SCK 控制之后，RCK 控制之前，因此 QH 寄存口每读取一次,QH’即输出一次 QH 的数据。

在 12 路阀岛电路板中两个寄存器串联，每个控制 6 路，第一个寄存器的 QH’ 接在第二个寄存器的 SI。两个寄存器共用同一个 RCK 和 SCK，保证同时输入输出。SCLR 始终高电平，G 始终接地。



寄存器每个寄存口的输出都按顺序对应光耦板输入端的对应焊盘。

1.5.3 软件设计

1.5.3.1 系统架构与运行流程

架构基本介绍

工程车作为本赛季的重要机器人之一，其基础框架仍使用 ARTINX 战队自主搭建的代码框架，不同于 RTOS 这种通用的操作系统，ARTINX 战队编写了一套专用于 Robomaster 系列比赛的代码框架，用来做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，更重要的是，**本队的代码框架上层均有 C++ 编写，所有数据结构都被抽象成了类，更符合机器人的设计要求，降低了开发者的上手难度。**我们抛弃了 RTOS 使用了“更新模式”这种设计模式来实现逻辑并行，并彻底重写了 CAN 通讯的底层收发机制实现了一套简易的通用数据包通信。

其核心创新点特点可以总结为以下几条：

- 类似游戏引擎更新模式的机器人逻辑控制框架。
- 简单的反射系统，防止在逻辑控制代码里单例满天飞。
- 功能完善的状态机框架。
- 与以往完全不同的，与硬件解耦的 CAN 总线通信机制。

- 完整的，基于数据包的通信框架；简单的比特流助手类，便于序列化反序列化数据包；使用观察者模式实现的 RPC。

核心代码介绍

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，消息处理类的注册。Systick 初始化之后开始执行控制程序。这里面包含了 STM32 原生的外设初始化，和代码框架内类变量的初始化。

```
int main(void)
{
    // NVIC setup. Remember, no Preemption.
    NVIC_PriorityGroupConfig(NVIC_PriorityGroup_0);

    Dr16::Instance()->Init();

    UnitreeRS485MotorCommander::Instance()->Init();

    unitree.Init();

    Time::Init(MS_PRE_TICK);

    SysTick_Init(MS_PRE_TICK);

    while (1)
    {
    }
}
```

Systick 中断

整个机器人的逻辑更新是基于 STM32 的 Systick 中断，在我们的代码框架中默认每毫秒执行一次，这个函数执行了挂墙时钟的更新和整个机器人逻辑的更新。

```
void SysTick_Handler(void)
{
    // Disable all interrupt
    __set_PRIMASK(1);

    Time::Tick();

    Dr16::Instance()->Update();

    UnitreeRS485MotorCommander::Instance()->Update();

    currentTime = Time::GetTick();

    unitree.Tick();
}
```

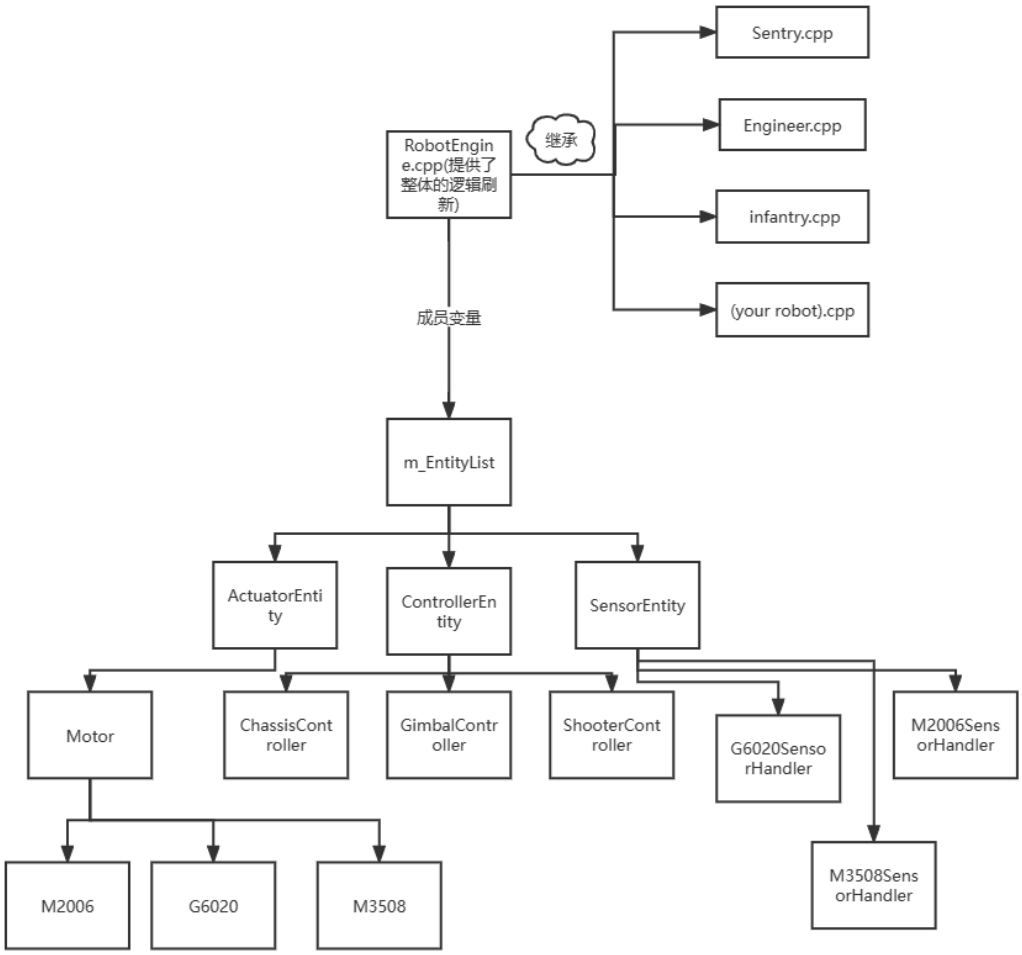
机器人更新

更新模式是游戏引擎最基本的设计模式，框架中由一个 RobotEngine 类的实例负责执行所有其拥有的 Entity 类进行 Update()操作，其中 Update()为虚函数，不同的 Entity 通过重载 Update()来实现不同的逻辑，从而实现逻辑并行。

RobotEngine 负责更新控制逻辑，原理类似游戏引擎的更新模式，一个 Robot 类需要继承自 RobotEngine 类，并且把需要更新的 Entity 聚合成 Robot 的成员变量，Entity 的构造函数中会根据 Enity 的种类注册到 RobotEnigne 管理的一个数组里面，聚合在别的 Entity 里的 Entity 也会自动注册到 RobotEngine 里，例如 M3508 类里面的 M3508Sensor 类。Tick()方法被调用的时候，会依次调用在 RobotEngine 里注册了的 SensorEntity 类、ControllerEntity 类和 ActuatorEntity 类的 Update()方法。同种类别的 Entity 在一次 Tick()里的调用顺序未定义，比如同为 ControllerEntity 的 GimbalController 和 ChassisController 的执行顺序是不确定的，但是在任何注册了的 ControllerEntity 更新之前，所有注册了的 SensorEntity 类都已执行完其 Update()方法，所以在 Controller 里读取传感器时，时效性能够保证。同样的，ActuatorEntity 执行时能保证所有的 ControllerEntity 都已更新完毕，防止出现指令滞后的情况。SensorEntity 参考 M3508SensorHandler 类，ControllerEntity 参考 ChassisController 类，ActuatorEntity 参考 M3508 类，RobotEngine 类参考 Testbot 类。

为了方便 Entity 之间通信，实现了简单的反射系统，能够运行时确定 Entity 的种类。

这就是整个机器人系统的更新逻辑，具体的流程拓扑图如下图所示：



代码拓扑图如下图所示：

```
class M3508Sensor : public SensorEntity
{
    // ...
    virtual Init();
    virtual Update();
    // ...
};

class M3508 : public ActuatorEntity
{
    // ...
```

```
    M3508Sensor m_Sensor;
    // ...
    virtual Init();
    virtual Update();
    // ...
};

class ChassisController : public ControllerEntity
{
    M3508 m_Motors[4];
    // ...
    virtual Init();
    virtual Update();
    // ...
};

class TestBot : public RobotEngine
{
    // ...
    ChassisController m_ChassisController;
    // ...
};

TestBot testbot;

void SysTick_Handler(void)
{
    // ...
    testbot.Tick();
    // ...
}
```

```
int main()
{
    // ...
    testbot.Init();
    // ...
}
```

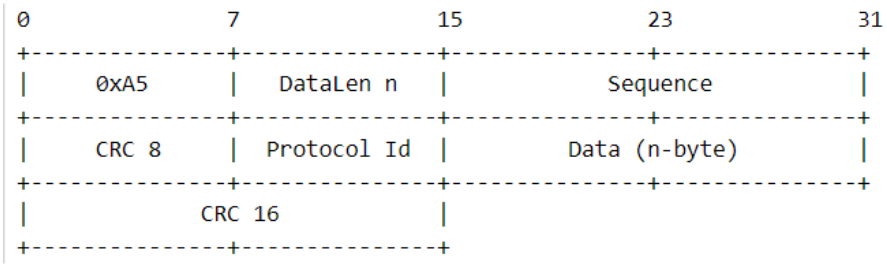
RobotEngine::Tick()，会遍历所有记录下来的 Entity，检查是否应该在当前执行其 Update()成员函数。Tick()应该每个 Systick 都调用一次。有三种类型的 Entity，分别是 SensorEntity，ControllerEntity 和 ActuatorEntity，每次调用 Tick()时，只有在所有 SensorEntity 都遍历了之后才会开始遍历 ControllerEntity，同样，遍历完所有 ControllerEntity 才会遍历 ActuatorEntity。相同类型的 Entity 之间执行顺序不确定。

具体来说：

每次在 SysTick 中断中，testbot.Tick();被调用的时候，会执行 4 个 M3508Sensor 的 Update()函数，更新电机传感器的反馈；ChassisController 的 Update()函数，根据控制逻辑控制 4 个电机的速度或是位置；以及 4 个 M3508 的 Update()函数，计算出电机具体的控制电流并且发送出去。

通讯设计

板件通讯，上下位机通讯，裁判系统通讯，我们同时采用了官方裁判系统的通讯模板，数据包格式如下图所示：



数据包是比特流，bool 类型只占一个比特，参考《网络多人游戏构架与编程》的设计，我们使用比特流，序列化，反序列化来压缩和传输数据。

至于消息的收发方式在现在的框架下核心原理可以总结为：

所有通过**外设收到的消息**，会被放入到一个循环队列中储存，每过一段事件，会清空循环队列中的

数据，并处理。理方式通过寻找提前存在 hash 表中的以处理 ID 为 key 值的信息处理器实现。

数据的发送则是所有节点的信息传递需求都被塞到一个 buffer 里，通过定时的发送来保证数据的通畅和外设的不堵塞。

如果你要用自定义的方法处理一个特定 CanId 的 CAN 数据帧里面的数据，你需要一个 CanMsgHandler 类子类的实例，override HandleNewCanRxMsg(CanRxMsg* _msg)函数，在里面实现处理数据的方法，并且注册到 CanMsgDispatcher::Instance()中，收到该 CanId 的 CAN 数据帧时，这个 HandleNewCanRxMsg(CanRxMsg* _msg)会在 Systick_Handler(void)中 CanMsgDispatcher::Instance()->Update()调用时被调用。示例如下：

```
    class CanImuDataReader : public CanMsgHandler
    {
        virtual void HandleNewCanRxMsg(CanRxMsg* _msg)
        {
            // 先调父类的处理函数，把消息复制下来
            CanMsgHandler::HandleNewCanRxMsg(_msg);

            // Read the IMU data, frame structure is documented at ******.pdf
            // ...
        }
    };

    CanImuDataReader canImuDataReader;

    int main()
    {
        // ...
        // 告诉 CanMsgDispatcher 在 CAN1 上接收到的 ID 位 0x250 的 CAN 消息给
canImuDataReader 去处理
        CanMsgDispatcher::Instance()->RegisterHandler(CAN1, 0x250,
&canImuDataReader);
        // ...
        // Start Systick Interrupt
        for(;;)
```

```
        {
            ;
        }
    }
```

CAN 消息发送只要调用：

```
    CanManager::Instance()->CanTransmit(CAN_TypeDef* _can, uint32_t _id,
uint8_t* _pData, uint32_t _len);
```

即可发送任意长度的字节数组，CanManager 会自动把数据分成 8 个字节一组的包，按顺序发送，并且确保每路 CAN 每个 Tick 最多发 3 个 CAN 数据帧。

我们的 CAN 消息接收如下图所示：

```
    void CAN1_RX0_IRQHandler(void)
    {
        CanRxMsg _rxMsg;

        if (CAN_GetITStatus(CAN1, CAN_IT_FMP0) != RESET)
        {
            CAN_ClearITPendingBit(CAN1, CAN_IT_FMP0);
            CAN_Receive(CAN1, CAN_FIFO0, &_rxMsg);
        }

        CanManager::Instance()->MsgQueue(0)->Enqueue(&_rxMsg);
    }
```

收到 CAN 消息进入 CAN 中断之后，不进行任何处理，仅仅是把收到的 CAN 消息完整地保存在了一个循环队列中。

```
    void CanMsgDispatcher::Update()
    {
        CanManager& _canManager = *CanManager::Instance();
        // 按顺序处理 CAN1 和 CAN2 的 CAN 消息
        for(int i = 0; i < 2; ++i)
```

```
        {
            // 清空队列
            while(!_canManager.MsgQueue(i)->IsEmpty())
            {
                // 将一个消息出列
                CanRxMsg& _rxMsg = *_canManager.MsgQueue(i)->Dequeue();
                // 在哈希表中寻找当前 CAN 消息 ID 所对应的 CanMsgHandler
                CanMsgHandler** _handler =
m_CanIdHandlerTable[i].Search(_rxMsg.StdId);

                if(_handler != nullptr)
                {
                    // 如果找到了，让这个 CanMsgHandler 自己去处理这个数据
                    (*_handler)->HandleNewCanRxMsg(&_rxMsg);
                }
            }
        }
    }
```

每个 SysTick_Handler 中，调用 CanMsgDispatcher::Instance()->Update()时，会一个一个地按顺序处理上一个 tick 和着一个 tick 之间接收到的 CAN 消息，从队列里拿出消息之后，用其 ID 在哈希表中查找之前注册了并且想要处理这个 ID 消息的 CanMsgHandler，如果找到了就用它去处理接收到的消息。

板间通讯，上下位机通讯同理。对于这类自定义包格式的数据，解析包的时候是一个一个字节的解析的，有一个 Header Buffer 与 Packet Buffer。如果收到了 0xA5，会把正在处理的字节同时放的 Header buffer 与 Packet buffer；当当前处理的字节正好距离上一个 0xA5 有 4 个字节的时候，会进行包头的 CRC8 校验；假如通过校验了，就把当前 Packet Buffer 的内容全清空，并且在收到由包头预期长度的数据之前不做任何操作；收到预期长度之后，进行 CRC16 校验；如果 CRC16 校验也通过了，那说明完整地收到了一个数据包，同样类似于 CanMsgDispatcher，收到包后将通过 ProtocolID 在一张哈希表里查询想要处理这个包的 MsgHandler，并且执行其对应的 OnPacketReceived()函数。

发送的时候也是有一个缓冲队列的，调用 packet 的 sendpacket 之后，数据不一定会立刻发出，有可能会等到串口空闲或是 CAN 空闲的时候立刻发送，只要确保发送带宽不超过其硬件的极限，程序会尽可能快地发送数据。

1.5.3.2 重要功能

作为本赛季重要的方案之一，机械臂的电控方案主要需要解决两个重要问题：

1. 如何依据机械臂的各种参数来进行电机选型和动力学方案设计
2. 如何设计运动方案来处理机械臂多自由度
3. 如何合理的设计人机交互方案，使操作手能在有限的键位条件下能依靠自动化简单的走完取矿，兑矿的流程，同时又保持一定的操作性来应对比赛场上的突发情况。

机械臂动力学方案设计

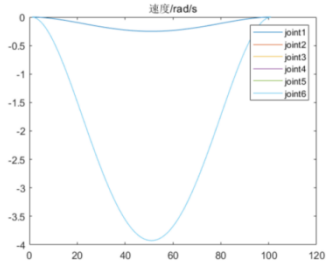
关节电机选型

鉴于我们设计的类 SCARA 构型的机械臂，我们的驱动器将不再负载整个末端负载的重量，关节驱动器只需提供瞬时加速所需的驱动力和对抗系统的动摩擦。这可以将我们的电机力矩最大利用到加速上，虽然损失了一定的的稳定性但在速度上，我们有绝对的优势。

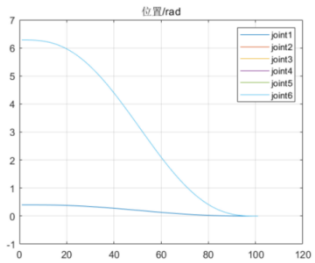
为了保证我们的运动性能，电机需要以一定的速度要求提供足够的扭矩。我们尝试对机械臂的工况进行动力学仿真，计算出电机所需要输出的功率。

对于六个负载电机我们分为姿态三轴和位置三轴，承担大负载的为关节 23 的转动关机和关节 5 保持负载支并撑重力的转动电机。未了保证末端移动的连续，我们采用运动学规划设计一套合理的加速度速度和角度曲线来控制电机。

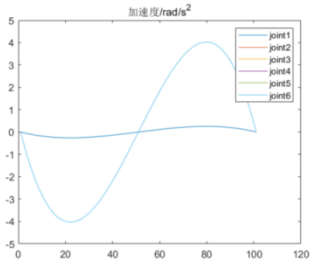
我们的工况选择了将每个关节一整周，并在末端负载 4kg，模拟矿石和末端执行器，整个流程在 3s 内完成，使用线性 5 次插值计算的加速度，速度和位置规划如下：



关节速度图



关节位置图

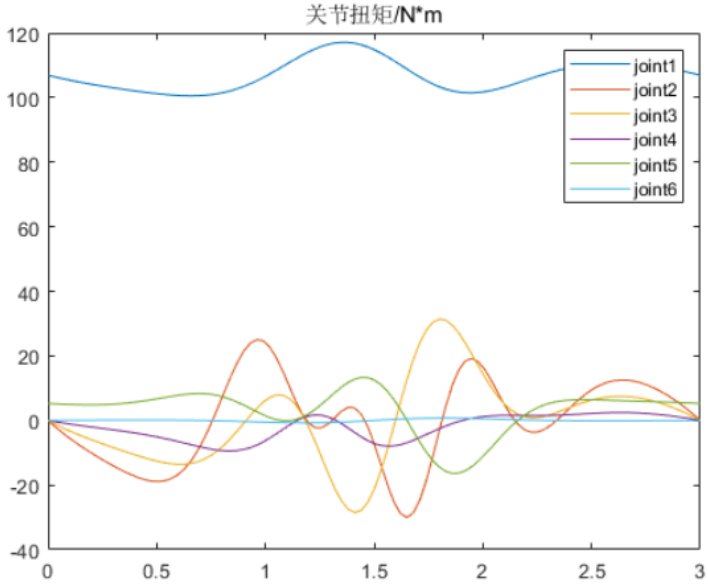


关节加速度图

由于我们的机械臂需要做到连续的坐标变化，所以我们关注于需调速的机械。选择几个关键字：负载，短期运行，断续运行，调速平滑程度高，无特殊场所，变功率负载特性。根据机械设计手册可得：**调速范**

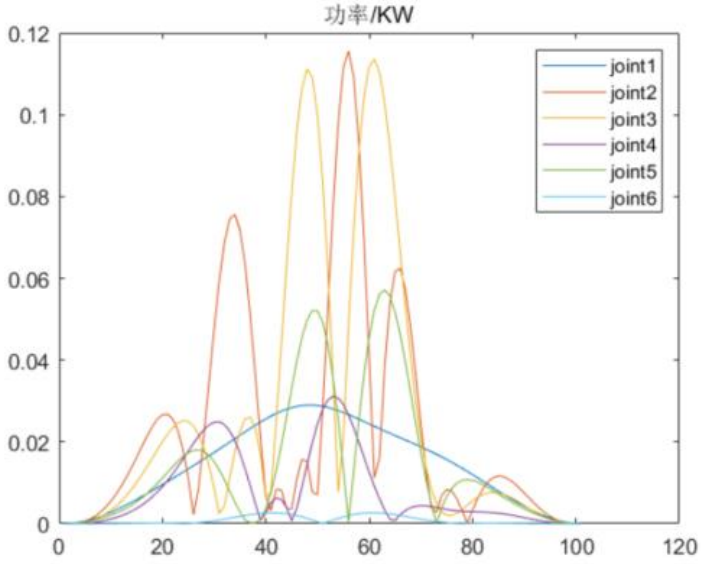
围 1：3 以上，需连续稳定平滑调速的机械，采用直流电动机。

有了上述的加速度速度和角度的规划曲线，我们很容易能得到六个电机的负载扭矩分布，如下图所示：



通过电机功率计算公式可得功率图如下：

$$p = \frac{T_b n_D}{9550}$$



我们选取的电机的额定功率应大于上述的最大功率。

同时查询机械设计手册可得对于直流电动机需要满足以下公式：

$$I_{MAX} \leq K\lambda_1 I_N$$

K——余量系数，直流电动机为 0.9-0.95

I_N——电动机额定电流

λ_1——允许电流过载倍数，对于直流电机一般为 1.5

通过上述式子我们可以计算出电动机的最大输出电流，把这个电流乘上电机特有的转矩系数就可以得到 最大转矩，这个最大转矩应该大于所有工作范围下的转矩。

根据我们的工况，我们最后选择变动负载连续周期工作制的校核

$$T_N = T_{MAX}0.9Kn\lambda T$$

通过排序得到六个关节的最大负载扭矩（力）如下所示：

T_1 = 117.0906N

T_2 = 25.0322NM

T_3 = 31.3977NM

T_4 = 2.4588NM

T_5 = 13.3176NM

T_6= 0.7343NM

根据之前选取的电动机转矩过载倍数 102.5 可以计算得到额定转矩

T_N2 应该大于 18.5423NM;

T_N3 应该大于 23.2576NM;

T_N4 应该大于 1.8214NM;

T_N5 应该大于 9.8649NM;

T_N6 应该大于 0.5439NM;

1. 对于末端第 456 关节，我们均选取 RobomasterM3508 直流无刷电机搭配不同的减速机，来实现工况的适配：

第 4 关节（第 6 关节同理）选取新的减速比为原装 1:19，其额定扭矩 TN 为 3NM 满足大于 1.8214NM。最大转速为 469rpm 转化为 49.09rad/s 大于 3.927rad/s。再对最大电流校核：电机能接受的最大电流为 Imax = IN * K * λI = 10 * 0.95 * 1.5 = 14.25A。在我

们的工况中，需要的最大扭矩为 2.4588NM，除以转矩常数得到电流为 8.196A 符合。

第 5 关节选取新的减速比为原装 1:71，其额定扭矩 TN 为 11.21NM 满足大于 9.8649NM。最大转速为 8.028rad/s 大于 3.927rad/s。再对最大电流校核：电机能接受的最大电流为 Imax = 14.25A。在我们的工况中，需要的最大扭矩为 13.3176NM，除以转矩常数得到电流为 11.87A 符合。

2. **对于第 23 关节，我们选择用宇树 GO-M8010-6 关节电机。**这两个关节的最大扭矩为 25.0322NM 和 31.3977NM，额定转矩应大于 18.5423NM 和 23.2576NM，对应的最大电流为 39.18A 和 49.14A。这些数值在 GO-M8010-6 关节电机的承受范围之内。

GO-M8010-6 电机

M3508 电机

3. **第一关节为平动关节，我们仍然使用 M3508 电机，搭配链传动。**在我们的动力学建模中，该平动关节的最大速度为 0.25m/s，最大加速度为 0.257m/s^2，抬升的总重为 5.9kg。

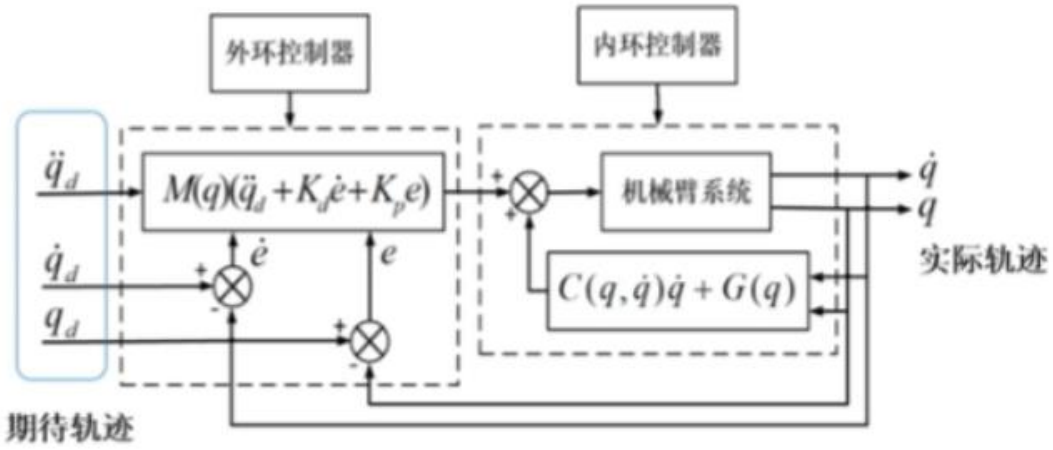
我们选用的链轮为 2 分 10 齿的链轮，传动半径为 10mm，额定扭矩为 3NM 的电机能输出 300N 的力，大于 T_N1，满足需求。

电机控制方式

对于末端三轴上的 3508 电机，他们所支付的力矩在轨迹规划的全过程上不会有巨大突变，且这些电机的旋转范围有限，并不是很需要前馈的力矩的补偿来保证电机反馈输入变化的平滑。所以我们仍使用传统的 PID 算法来做闭环控制

对于第 1 抬升关节的 3508 电机，这个电机始终受到单一方向向下的负载，这保证了电机驱动的稳定，且由于我们上面的计算可知，该电机的力矩变化范围也不大，所以该电机的控制方式也是用传统的 PID 算法。

对于第 2，3 关节的 GO 电机，这两个电机是传统的关节驱动。基于经验，我们尝试使用前馈+反馈的方式来对电机做控制。具体方式如下图所示：

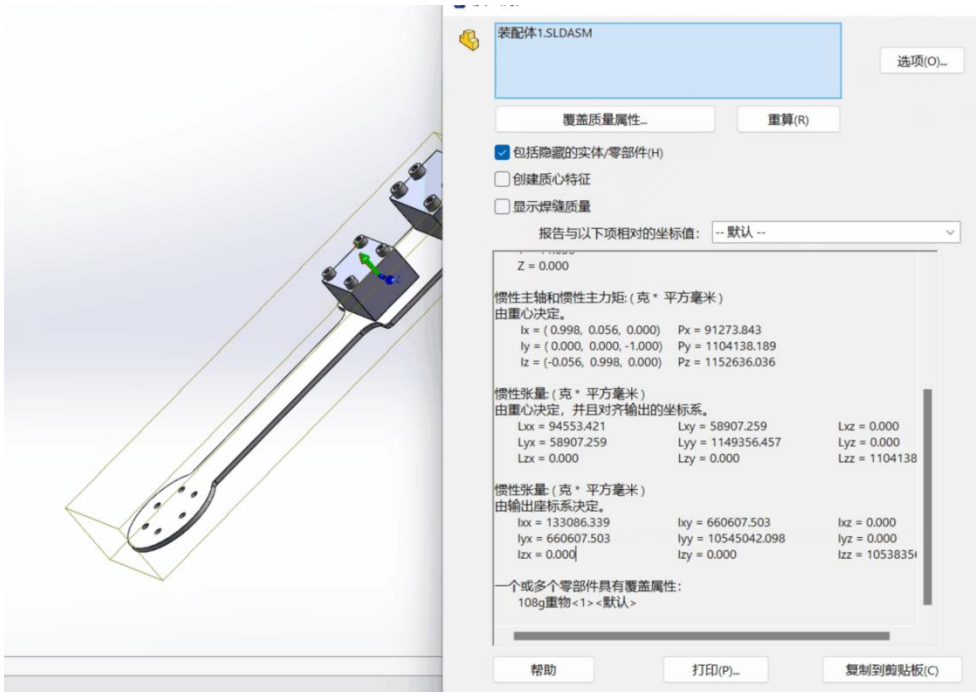


前馈力矩在整个控制系统的输入中占大部分，它的得出是通过解耦机械臂每个关节的运动。这种**非线性补偿器**能够很好的解决单负反馈解决不了的**奇点力矩突变**等问题，而反馈力矩只做补偿作用，在控制系统的输入部分只占小部分。

为了实现前馈力矩控制，我们做了以下实验：

基于最简单的模型，我们搭建了以下的实验系统：**单个电机驱动远端负载旋转**

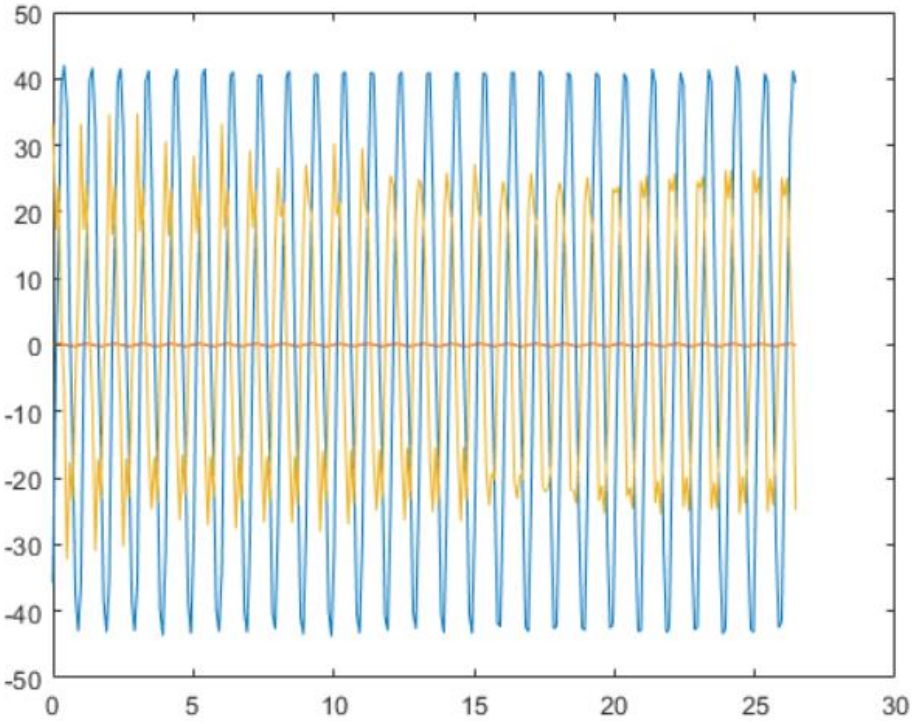
根据 Solidworks 的建模我们得出了该负载的转动惯量如下图所示：



基于物体的动力学属性，我们设计了正弦式的系统输入尝试做系统辨识，探究前馈力矩计算的可能性，对于模拟的系统，我们使用传统的二阶系统，构建方程如下:

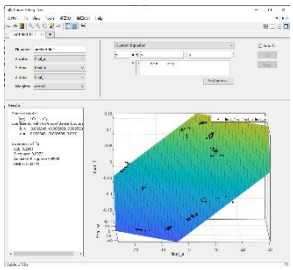
$$T = k_1 \times a + k_2 \times w$$

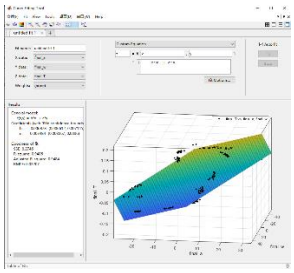
我们采集的数据如下图所示：

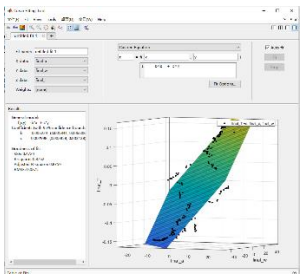


我们总共做了 40 次实验，每个条件 10 次，最后取平均值

拟合得到数据如下（节选）：







General model:
f(x,y) = b*x + c*y
Coefficients (with 95% confidence bounds
b = 0.00681 (0.006647, 0.006974)
c = 0.003485 (0.003384, 0.003587)

Goodness of fit:
SSE: 0.1643
R-square: 0.9697
Adjusted R-square: 0.9696
RMSE: 0.02467

General model:
f(x,y) = b*x + c*y
Coefficients (with 95% confidence bounds
b = 0.006598 (0.006317, 0.00688)
c = 0.003265 (0.003087, 0.003443)

Goodness of fit:
SSE: 0.2646
R-square: 0.9312
Adjusted R-square: 0.9308
RMSE: 0.03684

General model:
f(x,y) = b*x + c*y
Coefficients (with 95% confidence bounds
b = 0.006715 (0.006483, 0.006946
c = 0.003346 (0.003197, 0.003494)

Goodness of fit:
SSE: 0.3886
R-square: 0.9337
Adjusted R-square: 0.9335
RMSE: 0.03686

总结如下：

```
T = 1s Torque = 2.2155时

K1 = 0.0083,  K2 = 0.0057

拟合置信度 0.9321


T = 1s Torque = 1.899时

K1 = 0.0080,  K2 = 0.0052

拟合置信度 0.9742


T = 1s Torque = 1.266时

K1 = 0.0067,  K2 = 0.0034

拟合置信度 0.9507


T = 1s Torque = 0.9495时
```

```
K1 = 0.0063,  K2 = 0.0037

拟合置信度 0.8690
```

我们发现，随着驱动力的变大，该模型内部的参数变大，这就说明我们的模型过于简单表达不了电机的力矩。

从理论上分析

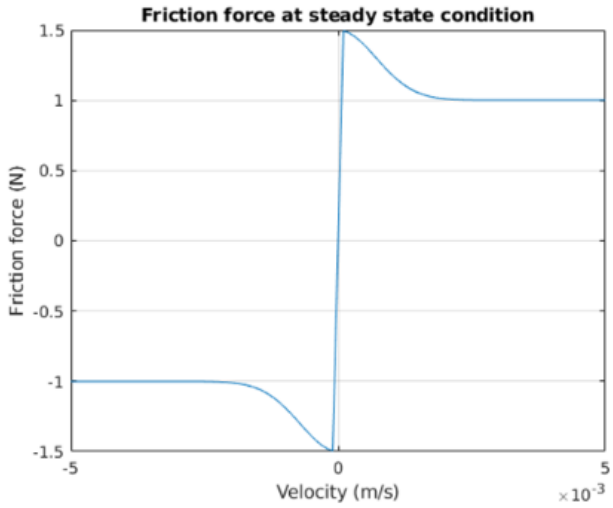
$$M = \alpha * 1$$

力矩等于转动惯量乘上角加速度，并不会有高阶项的出现，我们怀疑是摩擦力和系统自带阻尼导致模型不准，所以我们尝试补充模型：

比如加入 lugre 摩擦模型：

$$\frac{dz}{dt} = v - \sigma_0 \frac{|v|}{g(v)} z = v - h(v)z$$

$$F = \sigma_0 z + \sigma_1 \dot{z} + f(v)$$



v 是接触面的相对速度，z 是内摩擦状态，F 是预测的摩擦力

最后都没有得到好的效果。

考虑到由于我们的构型设计，该关节不会发生很大的力矩突变，只要设计好轨迹，保证加速度，速度连续就不会有太大的问题，所以我们又回归了传统的 PID 控制。

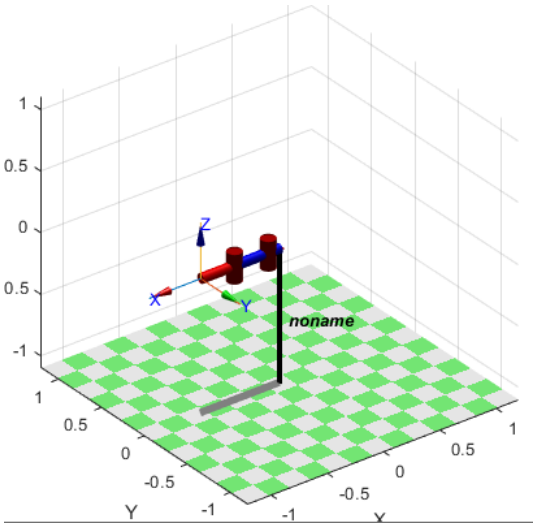
最后所有的关节都以相对平稳的方式运动。

机械臂运动学方案设计

我们将整个机械臂分为两个部分，前三轴决定了整个机械臂的位置，后三轴决定了整个末端的姿态。

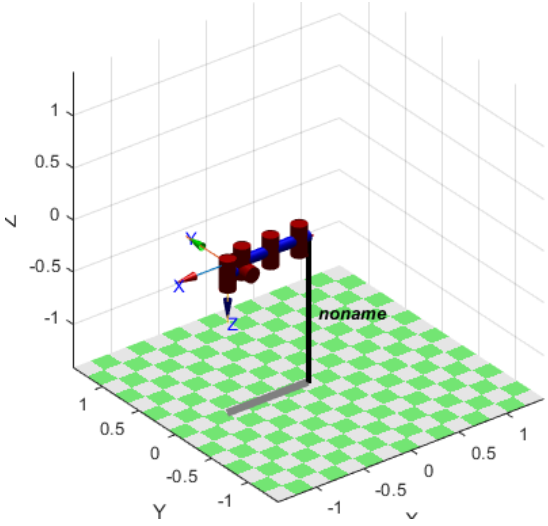
我们希望的是做到运动学解耦，位置三轴和姿态三轴隔离开。

如果我们尝试先决定姿态三轴，再决定位置三轴，那么位置三轴上电机的转动势必会引起姿态的变化。同理，如果我们先决定位置三轴，除非把姿态三轴的 RRR 构型的每轴偏移设计为 0，否则姿态的转化必定会引起位置的变换。



位置三轴构型图：PRR

但是，通过观察我们设计的位置三轴，我们发现，无论位置三轴怎么运动，影响的只有 z 轴方向的旋转角度，那么如果我们在位置三轴决定的情况下，给 z 轴方向的旋转增加一个角度修正，那么姿态三轴就与位置三轴独立。这里就涉及到我们的一个设计点，我们将末端 RRR 构型的第一个旋转关节设计的与位置三轴的两个旋转关节的旋转轴平行，如下图所示：

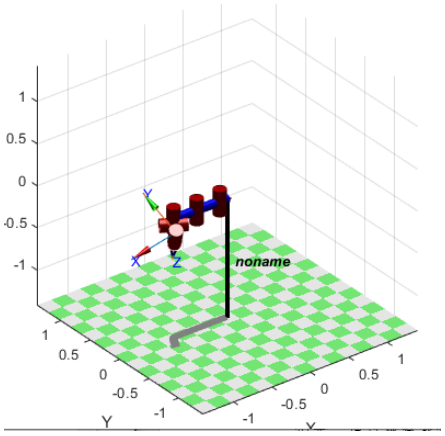


整体机械臂型图：PRR+RRR

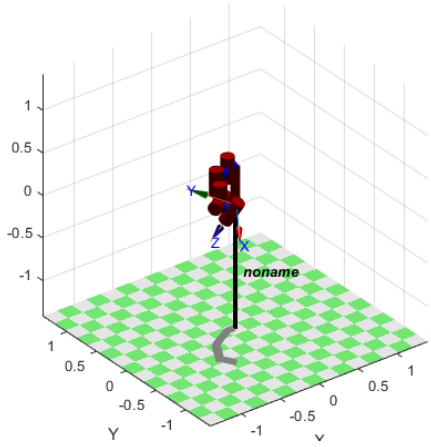
那么无论第 2 关节第 3 关节怎么旋转，第 4 关节的旋转轴都不会改变，因为们的旋转轴平行，如果不平行，那么我们提供调整补偿作用的关节的旋转轴就会偏转失去了补偿的能力。

具体解算流程如下：

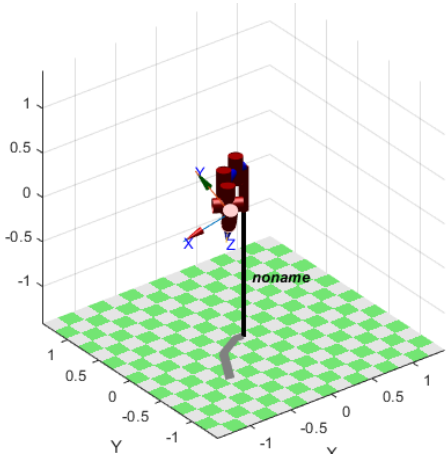
1. 根据所需要的姿态角反解出第 456 关节的角度。
2. 根据第 456 关节的角度，计算出位置偏移
3. 根据所需要的位置减去位置偏移得到关节 23 旋转的角度和关节 1 的伸长高度。
4. 最后旋转第 4 关节补偿由于位置三轴旋转带来的偏移。



步骤 1：计算姿态



步骤 2：计算位置

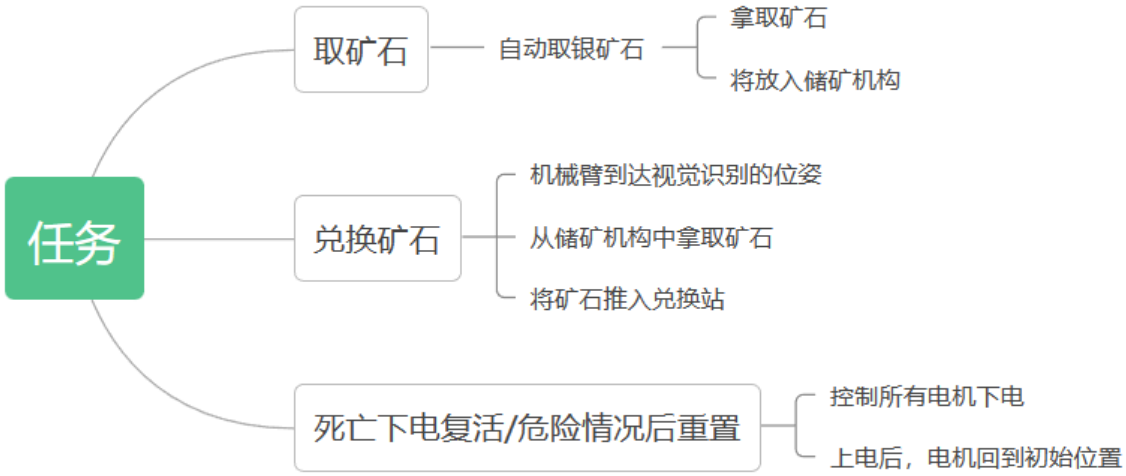


步骤 3：补偿姿态

这套计算流程将会是我们机械臂的主要控制框架和算法，在下一节我们人机交互会介绍一些其他算法（涉及运动结算）来补充这一节。

机械臂操作框架设计

我们首先需要明白我们需要实现的功能：



我们继续提出一些技术细节：

取矿石：

- 如何保证机械臂不与矿岛干涉？
- 如何保证在底盘每次到达银岛位置不同的情况下保证自动化能准确对准矿石？
- 如何保证机械臂在收回的时候不与机架干涉？

兑换矿石：

- 如何保证机械臂不与兑矿机构干涉？
- 如何控制矿石推入的角度？

重置：

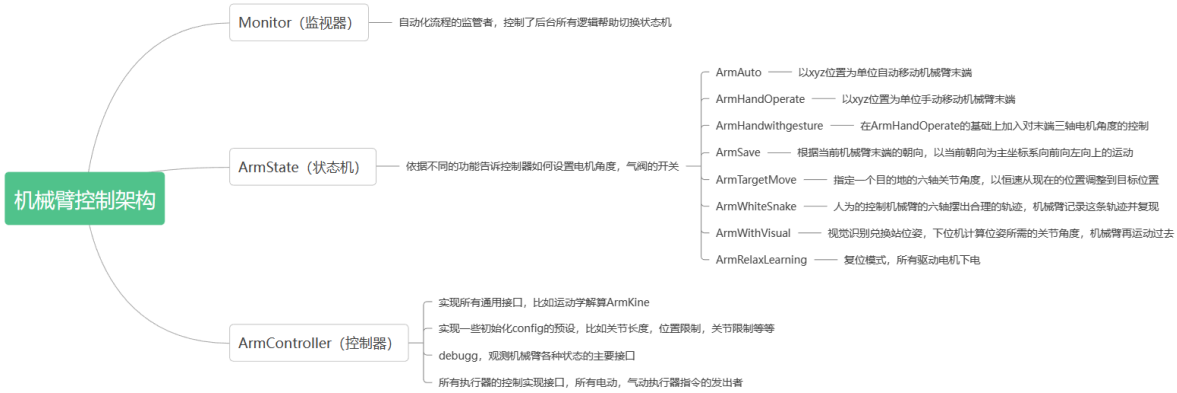
- 如何控制电机回到初始设定位置（本机械臂所有的电机均不含绝对编码器）？

解算兼容：

- 由上一节可知，我们的运动使用的是特定的解算，对于某一个特定位姿只能解出一个解，这就扼杀了机械臂的关节角组合的很多可能性。简单来说，如果我们单独控制每个电机摆出的那种姿态所对应的关节角组合，可能通过我们上述提出的算法解不出来，强行切换，就会导致同一位姿下关节的角度突变，那么如何切换纯解算运动和手动控制关节角度运动这两种模式就成了很大的问题。

针对上述的所有问题我们总共设计了 8 种状态机，1 种控制器，1 个监视器来解决我们的所有问题：

具体结构和各个组成成分的功能如下：



在介绍我们具体的自动化流程之前，我们首先介绍一下我们的动力学框架，其中涉及 13 个重要参数：

代表位置的三个变量 x_set，y_set， z_set。代表姿态的三个变量 link4_set， link5_set， link6_set(我们之所以使用这三个变量代表姿态，而不是 row，pitch，yaw，是因为直接设置 row，pitch，yaw 的变化是站在整个机器人坐标系下，而在操作者角度上这种变化是抽象的，而直接设置末端三个电机的旋转角度更加符合人的直觉)

代表电机转动角度的 7 个变量 q_1_set， q_2_set， q_3_set， q_4_set， q_5_set， q_6_set，q_4_set_bais，对于第四轴的特殊处理解决了我们上述的模式切换的问题，切换的核心矛盾在于第 4 关节的角度补偿，所以我们将这个关节提出来，独立处理，关节 4 的最终设定值 link4_set = q_4_set + q_4_set_bais。q_4_set 是由解算函数产生的，q_4_bais 则记录了人手操的过程，那么在模式且换的时候只要将当前的 link4_set 合理的分配到这两个值上就能避免解算导致的数值突变。

具体一点：我们将所有 8 个状态机分为两类，不使用解算函数,直接设置角度值(赋值 q_4_set)或者遥控器控制(赋值 q_4_bais)和使用解算函数设置 link4_set(赋值 q_4_set)。

那么这两类状态机之间的相互转化只需要执行两个函数来将 link4_set 合理分配到 q_4_set 和 q_4_bais 上就能解决所有问题。

在宏观角度上，我们使用到解算算法计算角度的状态机有 4 个 ArmHandOperate，ArmHandwithgesture，ArmAutoMove，ArmSave，使用上述算法，我们能始终保持机械臂末端姿态保持不变，因为 2，3 关节导致的姿态改变都会被第 4 关节补偿，而在另一类模式中改变的姿态，都被认为是 bais，不影响整个解算流程。

为了能对之后的自动化流程有更好的理解，这里我们具体介绍一些状态机：

ArmHandwithgesture

这个状态机包括了位置的计算和姿态的计算

位置计算使用之前提出的函数

```
static float* inverse_kine(float HT_Mat[4][4] , const float link[6] ,
const float lift_const, bool is_pick)
```

输入为目标 DH 矩阵，6 个关机长度，第一关节的丝杠传动比，以及解算过程中的多解脏标记。我们在这个环节中使用的 DH 矩阵使用的参数只有位置三轴，姿态三轴均为正方向，即旋转 3*3 分量为单位矩阵。这就保证了如果 m_4_bais 无值，无论位置怎么变化，末端始终朝前正方向，如果 m_4_bais 有值，无论怎么变化，末端都朝一个方向。

姿态计算则直接使用遥控器控制角度，代码如下所示：

```
link4_set -= Dr16::Instance()->GetLHAxis() * 0.002f;
link5_set += Dr16::Instance()->GetLVAxis() * 0.002f;
link6_set += Dr16::Instance()->GetRHAxis() * 0.002f;
```

这些设置值都是放在偏执项上，并不会影响解算。

ArmSave

这个状态机是为了处理兑换过程中推的最后一下，为了保证我们最后推的方向是末端执行器的朝向，我们则需要做正运动学。

我们定义六个关节的旋转矩阵如下图所示：

```
syms a2 a3 d1 theta1 theta2 theta3 theta4 theta5 theta6
T01 = [ 1          0     0           0;
        0          1     0           0;
        0          0     1           d1;
        0          0     0           1];

T12 =[  cos(theta2) -sin(theta2)    0   215*cos(theta2);
        sin(theta2) cos(theta2)     0   215*sin(theta2);
        0          0              1   0;
        0          0              0   1];

T23 =[  cos(theta3) -sin(theta3)    0   215*cos(theta3);
        sin(theta3) cos(theta3)     0   215*sin(theta3);
        0          0              1   0;
        0          0              0   1];
```

```
T34 = [ cos(theta4) 0              -sin(theta4)     0;
        sin(theta4) 0              cos(theta4)    0;
        0          -1              0               l4;
        0          0              0              1];

T45=[   cos(theta5) 0              sin(theta5)      0;
        sin(theta5) 0              -cos(theta5)     0;
        0          1             0               0;
        0          0              0              1];
T56=[   cos(theta6) -sin(theta6)    0               0;
        sin(theta6) cos(theta6)     0              0;
        0          0              1              100;
        0          0              0              1];
```

最后计算得到的末端 DH 矩阵如下：

T =

$$\begin{pmatrix} \cos(\theta_5)\cos(\theta_6)\,\sigma_1 - \sin(\theta_6)\,\sigma_2 & -\cos(\theta_6)\,\sigma_2 - \cos(\theta_5)\sin(\theta_6)\,\sigma_1 & \sin(\theta_5)\,\sigma_1 & 215\cos(\theta_2) + 215\cos(\theta_2)\cos(\theta_3) - 215\sin(\theta_2)\sin(\theta_3) + 100\sin(\theta_5)\,\sigma_1 \\ \sin(\theta_6)\,\sigma_1 + \cos(\theta_5)\cos(\theta_6)\,\sigma_2 & \cos(\theta_6)\,\sigma_1 - \cos(\theta_5)\sin(\theta_6)\,\sigma_2 & \sin(\theta_5)\,\sigma_2 & 215\sin(\theta_2) + 215\cos(\theta_2)\sin(\theta_3) + 215\cos(\theta_3)\sin(\theta_2) + 100\sin(\theta_5)\,\sigma_2 \\ -\cos(\theta_6)\sin(\theta_5) & \sin(\theta_5)\sin(\theta_6) & \cos(\theta_5) & d_1 + l_4 + 100\cos(\theta_5) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

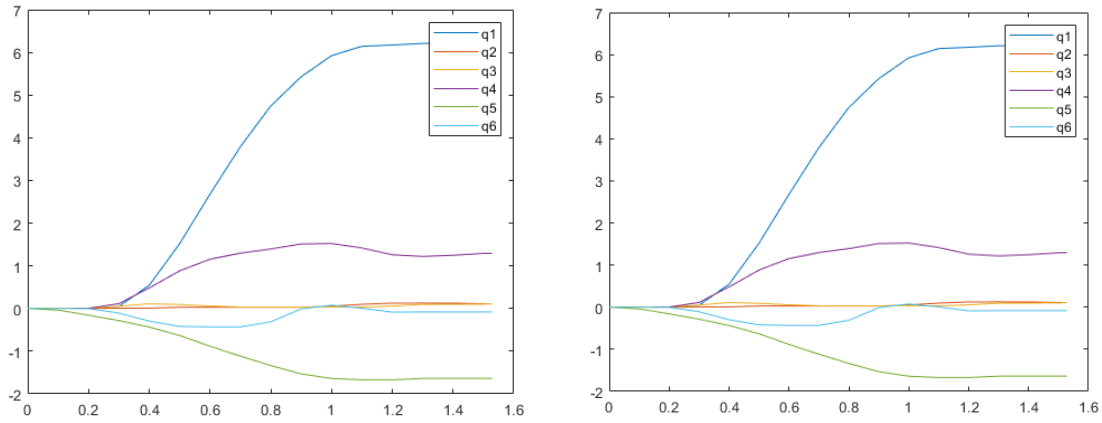
$$\sigma_1 = \cos(\theta_4)\sigma_3 - \sin(\theta_4)\sigma_4$$
$$\sigma_2 = \cos(\theta_4)\sigma_4 + \sin(\theta_4)\sigma_3$$
$$\sigma_3 = \cos(\theta_2)\cos(\theta_3) - \sin(\theta_2)\sin(\theta_3)$$
$$\sigma_4 = \cos(\theta_2)\sin(\theta_3) + \cos(\theta_3)\sin(\theta_2)$$

于是我们需要找到这是的 z 轴方向，并使用这时的 z 轴在世界坐标系下的 xyz3 轴的分量来进行控制：

```
x_bais = Dr16::Instance()->GetLVAxis() * 0.4f * (rot_matrix[0][2]);
y_bais = Dr16::Instance()->GetLVAxis() * 0.4f * (rot_matrix[1][2]);
z_bais = Dr16::Instance()->GetLVAxis() * 0.4f * (rot_matrix[2][2]);
```

ArmWhiteSnake

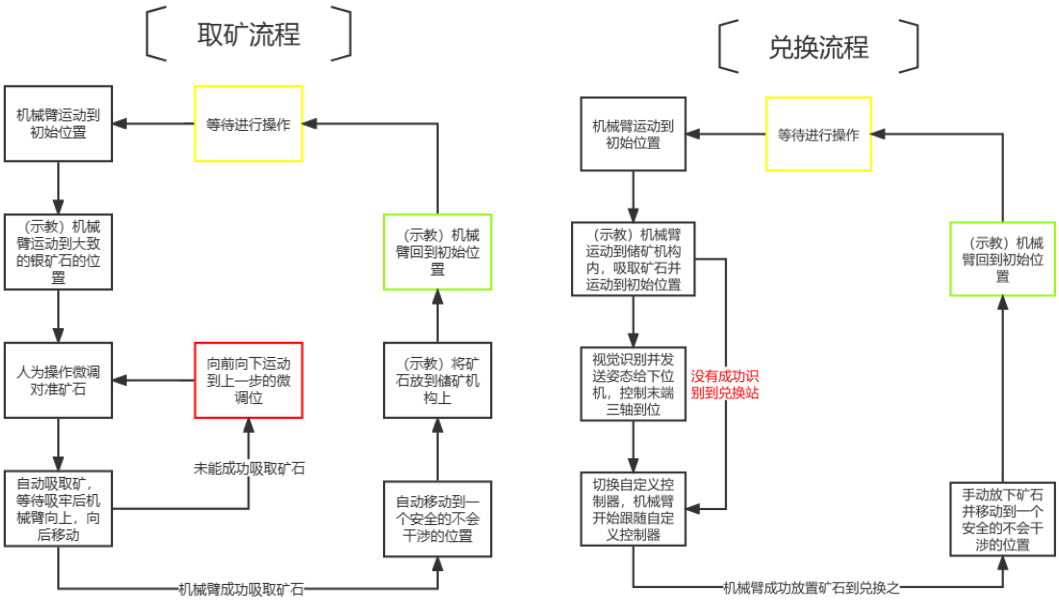
这个状态实现对人为操作姿态的复现，我们使用插值的方式用离散的点复现了整条轨迹，具体效果如下图所，左图为记录的六个关节角度的曲线，右图为重现的曲线：



可见效果十分优秀。

我们最后来介绍我们主要的两套自动化流程：

取矿流程，我们将这个流程抽象成几个步骤，如下图所示



这里我们解释一些技术细节：

在取矿的时候，我们没有将所有取矿流程交给机械臂的自动化。是因为，每次底盘到达矿岛的位置不一样，这就导致了虽然示教了机械臂但是它到达的位置任会有误差。由于我们的取矿方式是自下而上吸取矿石的顶部，那么底盘高度不变，产生偏移的只会有 x，y 两个坐标。因为 z 轴坐标准确，在给定一定裕度的情况下，一定不会发生干涉。那么这步的自动化＋人操微调的设计是十分合理的。

在取矿完成的时候，自动向上移动+向后移动是为了保证之后的移动不与兑换站发生干涉，移动到安

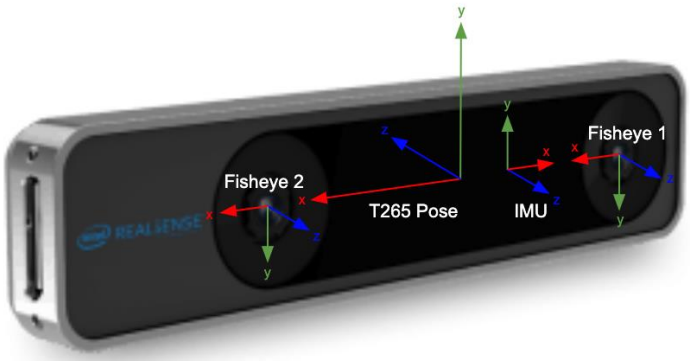
全位置，是为了统一所有的取矿流程，让把矿放入储矿机构这个步骤可以通过示教完成。后面的兑矿流程的相同操作也是由于这个原因。

在兑换的过程中，我们会主动将矿石放置在边缘，以防遮挡相机视野。在这个过程中，我们会通过 UI 给予操作手一些提示，相机是否识别到了目标兑换站。在这个过程中，操作手随时都可以选择终止，如果在下达终止指令的最后 1s 内，我们识别到了兑换站，末端三轴则会变形，如果没有识别到兑换站，末端三轴则不会变形，直接进入到自定义控制器的模式。

自定义控制器

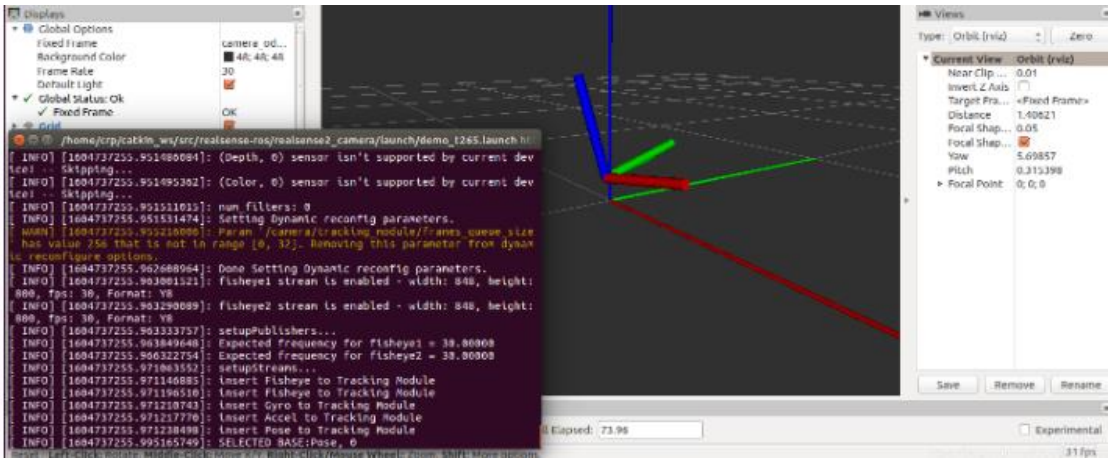
目的：这个赛季我们选择使用自定义控制器的方案来提升我们整体兑换的速度，传统控制器只能通过拨杆分别控制机械臂的六个自由度，而使用自定义控制器的方式能够实现手势追踪，保证操作手能以舒适的姿势同时控制六个自由度完成兑换的操作。

方案：为了实现这个功能，最简单的方法是建立起控制器和机械臂的六轴位姿关系，并将控制器和操

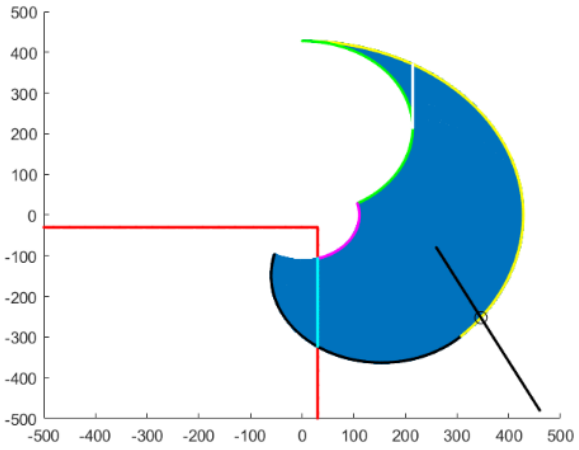


作手的手腕绑定起来，创造一个映射关系。简单的方案包括陀螺仪获取精确姿态，使用并联机构或者位移台获取位置，虽然并联机构能在一定程度上解决位移台占位置，阻挡手部操作的问题。考虑到机械研发成本，我们选择使用视觉 SLAM 的成品解决方案来实现对控制器位姿的获取。

T265 采用了 Movidius Myriad 2 视觉处理单元（VPU），V-SLAM 算法都直接在 VPU 上运行可直接输出 6DOF 相机位姿。我们在一台 NUC 上配置了 ROS 的相关环境，安装了 realsense 和位姿变化的基本支持包，这样我们就能很轻松的获取相机相对于开机位置的位姿变化。



我们再通过上位机将数据包下发到裁判系统，实现自定义控制器与下位机机械臂的通讯，整体数据包包括，位置三轴的偏移量和姿态三轴对应的欧拉角。



接着我们需要将相机坐标与机械臂坐标进行一个对应，由于我们的姿态三轴采用的是传统的球型手腕的构型，这就能保证三个驱动关节不互相影响，所以很自然的把三个欧拉角直接作为驱动关节的角度进行变化。至于位置三轴，我们将 z 轴坐标进行量纲转化赋给抬升关节电机，至于 xy 坐标，由于我们采用的是两个耦合度很高的旋转关节，所以需要做一些数据处理来保证获取的位置没有超出我们的工作空间。

我们的工作空间如上图所示，这里的限制包括两部分和机体本身碰撞或者关节角超限，为了避免关节角解算中奇异点的出现，我们选择只选取一种 elbow 构型（关节交点）即，两关节满足钝角关系，所以会出现 y 轴正方向的真空区，这在我们的兑换过程中是不需要的，同时另一种构型也会导致机械臂会比末端在 x 轴方向更靠近兑换站，这样会带来干涉的可能。所以上述工作空间为当前构型的最优解。

为了保证接收坐标的连续，我们做了以下操作，我们选择了一个相对中心的极点，当接收的上位机坐标超过了机械臂的工作空间，我们将以极坐标的方式，做一条超出点与极点中心的连线，并将极点连线与

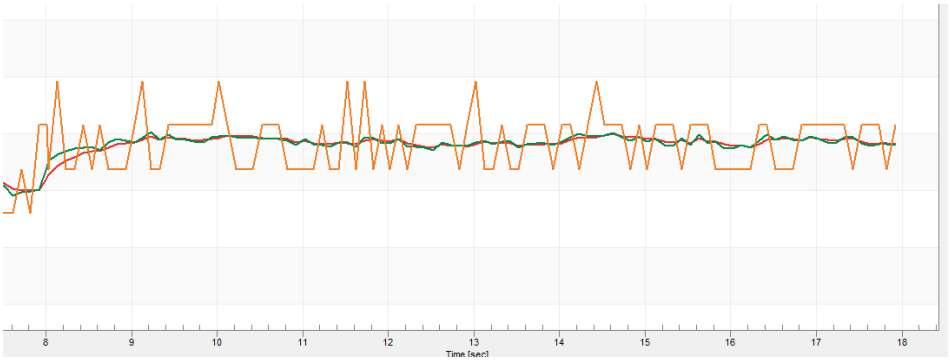
工作空间外围的交点作为实际移动点，这样上位机下发的坐标无论是否在工作空间范围内都能满足有一个实际运动点预期对应，这样的处理除了保证运动的连续性，同时解决了突变问题。如果没有这一步操作，如果出工作空间的坐标与回工作空间的坐标不一样，这就会导致坐标突变，从而导致机械臂抖动。

1.5.3.3 软件测试

关节电机控制

在分区赛回来之后，整体运动过程中惯性大的问题还是没有得到有效解决，为了改善这个问题，我们打算在三个方向上更新。首先肯定是整体机械结构的刚度，其次是电机控制，最后是运动规划。

1.对于电机控制，我们的改进有两方面，首先是对于 GO 电机我们得到的传感器的值并不是稳定的值，结果如下图所示：



所以我们需要使用滤波的方式在保留数据真实的情况下保证稳定性。

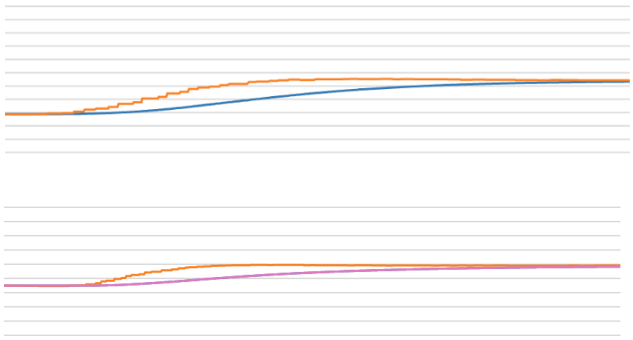
我们在分区赛版本采用的是一阶低通滤波，在国赛版本，我们则采取卡尔曼滤波。

```
public:
    void SetInput(float in){m_Input = in;}
    void SetTau(float tau){m_Tau = tau;}
    void SetResult(float out){m_Out = out;}
    void SetUpdatePeriod(float t){m_UpdatePeriod = t * 0.001f;}/* t in ms */

    float GetResult(){return m_Out;}
    float GetTau(){return m_Tau;}
    float GetUpdatePeriod(){return m_UpdatePeriod;}

    void Init(RobotEngine* _pOwner)
    {
        m_pOwner = _pOwner;
        m_LastUpdateTick = m_pOwner->GetTick();
        m_UpdatePeriod = 0.0f;
    }

    void Update()
    {
        m_UpdatePeriod = (m_pOwner->GetTick() - m_LastUpdateTick) * 0.001f;
        m_LastUpdateTick = m_pOwner->GetTick();
        float a = m_UpdatePeriod / (m_Tau + m_UpdatePeriod);
        m_Out = (1 - a) * m_Out + a * m_Input;
    }

    void Clear()
    {
        m_Out = 0;
        m_Input = 0;
        m_LastUpdateTick = m_pOwner->GetTick();
    }
};
```

一阶低通滤波的代码如下：

```
#ifndef MATH_FIRST_ORDER_FILTER_H
#define MATH_FIRST_ORDER_FILTER_H

#include <stdint.h>
#include <math.h>
#include "RobotEngine.hpp"

2384447291, 4 months ago | 1 author (2384447291)
class FirstOrderFilter
{
private:
    float m_Input;
    float m_Out;
    float m_Tau;
    uint32_t m_LastUpdateTick;
    float m_UpdatePeriod;

    RobotEngine* m_pOwner;
```

其中需要我们调节的参数为时间常数 tau，tau 的值越大，滤波后的结果越平滑，但是其动态响应也会变得更差，反应到图像上就是跟随性更差，如下图所示：



第二张图的 tau 值更大，所以需要更长的时间到达原始真实值。

另一种方法是卡尔曼滤波，其代码如下：

```
#ifndef MATH_KALMAN_FILTER_H
#define MATH_KALMAN_FILTER_H

#include <stdint.h>
#include <math.h>

// Q越大,跟的越紧密，滤波后的噪声越大
// R越大,即滤波后的数据跳动越小滤波输出收敛的越慢。
class KalmanFilter
{
private:
    float x_last;
    float p_last;
    float R;
    float Q;


public:
    void Init(float ProcessNoise_Q,float MeasureNoise_R)
    {
        R = MeasureNoise_R;
        Q = ProcessNoise_Q;
    }

    void Update(float ResrcData)
    {
        float x_mid;
        float x_now;
        float p_mid ;
        float p_now;
        float kg;

        x_mid=x_last;
        p_mid=p_last+Q;
        kg=p_mid/(p_mid+R);
        x_now=x_mid+kg*(ResrcData-x_mid);
        p_now=(1-kg)*p_mid;
        p_last = p_now;
        x_last = x_now;
    }

    void Clear()
    {
        x_last = 0.0f;
        p_last = 0.0f;
    }

    float GetResult(){return x_last;}
};

#endif
```

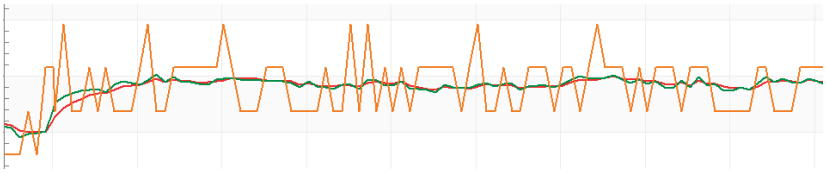
直观来看： kg 值越大，输出值越靠近新输入，kg 值越小，输出值越靠近过去的输入值（即更稳定，不趋于变化）。而 kg 值的大小决定自参数 R 和 Q：

R 是测量噪声，R 越大表示测量的误差越大，kg 值越小，输出自然更相信原来的值，而不是新的输入。

Q 是过程噪声，Q 越大，kg 越大，输出更跟随新的输入，但是在后续的计算中，越大的 kg 值导致了更小的 P_last 值，那么下一次计算中的 kg 自然就会变小。

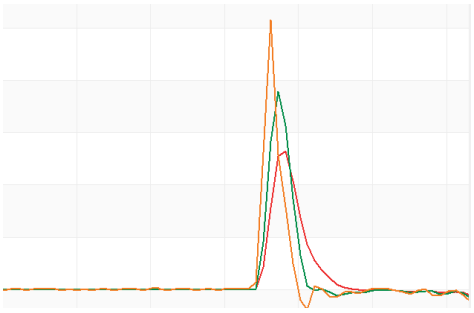
总结下来：Q 越大，跟的越紧密，滤波后的噪声越大；R 越大，即滤波后的数据跳动越小，滤波收敛的越慢。

我们将两种滤波方式横向对比得到以下结果：



黄色曲线为原始数据，绿色数据为卡尔曼滤波结果，红色数据为一阶滤波结果。可见两者都有很好的消抖效果，且基本上看不出差别。

但是在响应速度上，卡尔曼滤波表现出来更好的效果如下图所示：

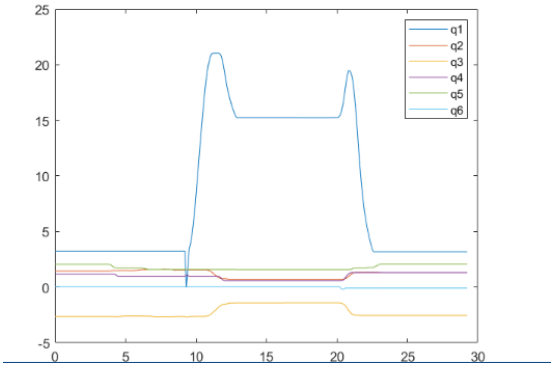


所以我们最终选择了卡尔曼滤波对我们的所有传感器数据进行处理。

另一个方面是尝试使用前馈的方法来实现关节角的突变，虽然在运动规划的前提下，我们避免了这些问题，但是有了前馈项，能从某种情况下摆脱反向运动带来的误差。

运动规划

我们的机械臂有两种控制方式，取矿完全由自动化实现，兑换由自定义控制器实现，我们采取示教的方式规划取矿的路径，并使用插值的方式对录好的关节角变化进行修改，来保证过程的流畅和避免干涉。



1.5.4 算法设计

在新赛季中兑换站拥有了更多的自由度，由于矿石与兑换站之间的间隙容差较小，矿石必须以相对准确的姿态放入兑换站中。本赛季中我队工程机器人装备了机械臂，为了充分发挥机械臂的作用，使用三级以上的兑换级别获取更高的经济效益，我队向工程机器人上安装了一套 NUC 系统，使用工业相机和视觉算法来识别兑换站的位姿。

1.5.4.1 算法简介与流程

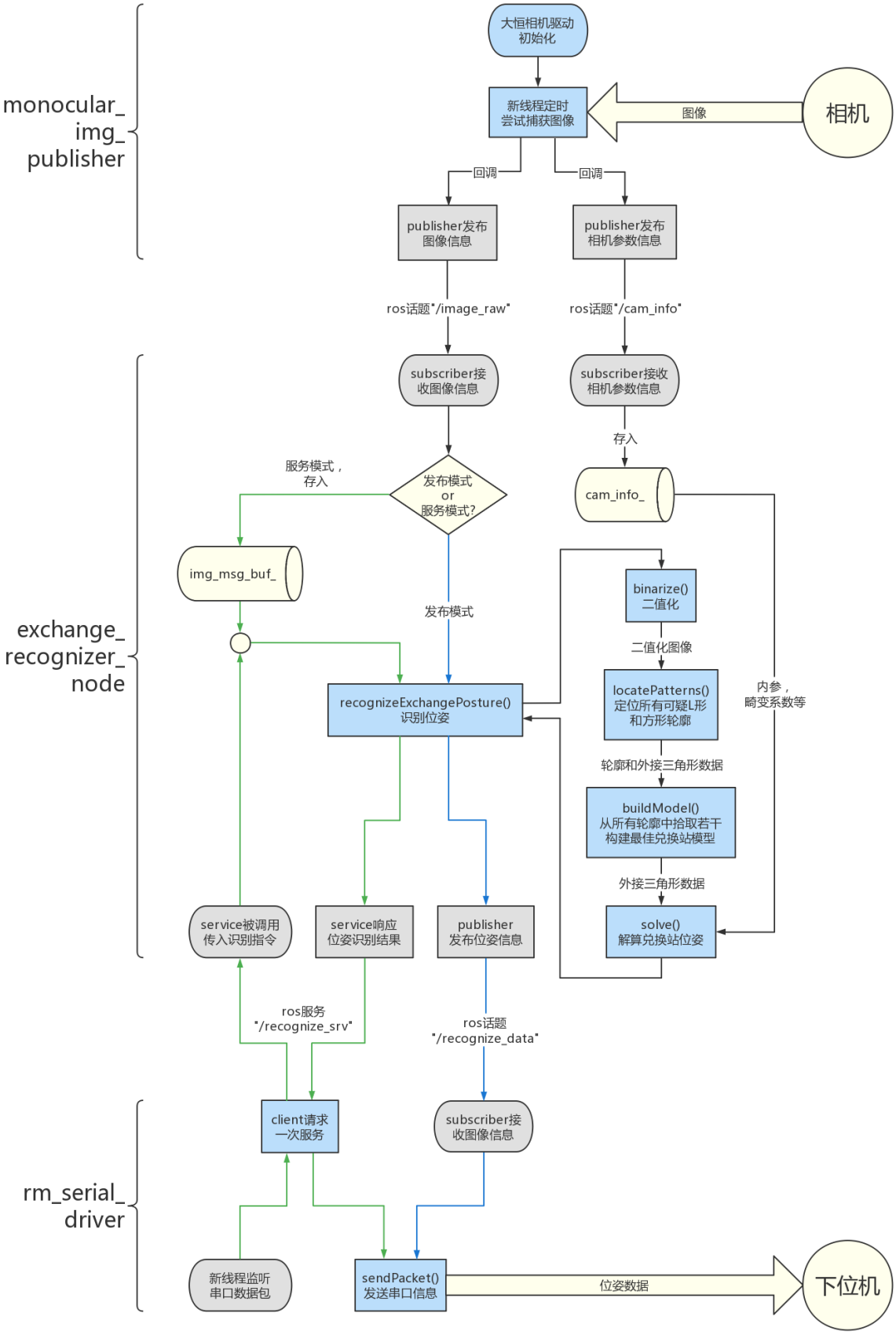
运行环境：NUC 一台，需安装 Ubuntu 20.04 系统，Eigen3 线性代数库，ros2 系统(foxy 版本)，大恒图像相机 SDK，以及 ros2 的 serial_driver 和 image_transport 插件。

视觉算法在 NUC 上安装的 ROS2 系统上运行，包含 3 个节点和若干调试用节点。程序运行时需要接收相机传入的图像，对图像进行二值化，筛选出合适的 L 形图块并定位 9 个参照点对应的二维投影点，使用 solvePnP 算法解算出三维位姿，经过转换后使用 usb 转 ttl 串口发送给下位机。由于该算法表现良好，现阶段暂时忽略第四个 L 形图块和两个方形图块。

程序包含 3 个节点：

- 单目相机画面发布节点 monocular_img_publisher
- 位姿识别节点 exchange_recognizer_node
- 串口通信节点 rm_serial_driver

程序流程图如下：


monocular_
img_
publisher
大恒相机驱动
初始化
新线程定时
尝试捕获图像
图像
相机
回调
回调
publisher发布
图像信息
publisher发布
相机参数信息
ros话题"/image_raw"
ros话题"/cam_info"
exchange_
recognizer_
node
subscriber接
收图像信息
subscriber接收
相机参数信息
存入
服务模式，
存入
发布模式
or
服务模式?
cam_info_
img_msg_buf_
发布模式
binarize()
二值化
二值化图像
recognizeExchangePosture()
识别位姿
locatePatterns()
定位所有可疑L形
和方形轮廓
内参，
畸变系数等
轮廓和外接三角形数据
buildModel()
从所有轮廓中拾取若干
构建最佳兑换站模型
外接三角形数据
service被调用
传入识别指令
service响应
位姿识别结果
publisher
发布位姿信息
solve()
解算兑换站位姿
ros服务
"/recognize_srv"
ros话题
"/recognize_data"
rm_serial_
driver
client请求
一次服务
subscriber接
收图像信息
新线程监听
串口数据包
sendPacket()
发送串口信息
位姿数据
下位机

队内目前使用大恒图像的工业相机拍摄画面，monocular_img_publisher 节点首先将大恒相机驱动初始化，而后从本包的 share 文件夹中读取相机内参，在一个新线程中定时尝试捕获图像，如果成功就使用 publisher 发送一个 ros 图像消息。

exchange_recognizer_node 节点有两种运行模式：服务模式和发布模式。

在 subscriber 成功接图像发布节点发来的图像后，如果是发布模式，就解析消息，使用 recognizerExchangePosture()函数识别位姿，而后把位姿信息通过话题“/recognize_data”上的 publisher 发给串口通信节点。rm_serial_driver 节点的 subscriber 接收到信息后使用 sendPacket()向下位机发送数据包。

如果是服务模式，识别节点就把消息缓存；串口通信节点会在新线程中监听串口数据，一旦收到下位机发来的包，就使用一个“/recognizer_srv”服务的客户端发送请求；识别节点收到请求后，再解析消息，使用 recognizerExchangePosture()函数识别位姿，而后把位姿信息作为响应数据发回串口通信节点，而后串口通信节点把位姿数据发送给下位机。

1.5.4.2 程序文件结构

```
.
├── README.md : readme 文件
├── rvizConfig.rviz : rviz 串口布局配置文件
└── src : 源码文件夹
    ├── engineer_bringup : 工程机器人的配置与启动节点
    │   ├── CMakeLists.txt
    │   ├── config
    │   │   └── engineer_bringup.yaml : 工程机器人的各节点启动配置，不包括 rm_serial_driver 的配置
    │   ├── launch
    │   │   └── engineer_bringup.launch.py : 启动工程机器人各节点的程序
    │   └── package.xml
    ├── exchange_posture_recognizer : 位姿识别节点
    │   ├── CMakeLists.txt
    │   ├── include
    │   │   └── exchange_posture_recognizer
    │   │       └── recognizer_node.hpp : exchage_recognizer_node 的源码头文件
    │   ├── package.xml
    │   └── src
    │       └── recognizer_node.cpp : exchage_recognizer_node 的源码
    ├── interfaces_msg : 一些自定义消息和服务的接口
```

```
│   ├── CMakeLists.txt
│   ├── msg
│   │   ├── AimControl.msg
│   │   ├── EngineerControl.msg
│   │   └── RecognizeData.msg : 发布模式下识别节点所发布的识别消息
│   ├── package.xml
│   └── srv
│       └── Recognize.srv : 服务模式下的识别服务
├── monocular_img_publisher : 相机图像流发布节点
│   ├── CMakeLists.txt
│   ├── config
│   │   ├── camera_calibration : 使用 opencv 算法标定的相机内参数据
│   │   │   ├── xxxxxxxxxxxxxx.xml : 一个相机对应一个“S/N 号.xml"文件
│   │   │   ...
│   │   └── camera_param.xml : 相机节点的参数文件，目前未使用
│   ├── include
│   │   └── monocular_img_publisher
│   │       └── monocular_img_publisher.hpp : 相机节点源码头文件
│   ├── launch
│   ├── package.xml
│   └── src
│       └── monocular_img_publisher.cpp : 相机节点源码文件
├── rm_serial_driver : 串口通信节点
│   ├── CMakeLists.txt
│   ├── config
│   │   └── serial_driver.yaml : 串口通信节点的配置文件
│   ├── include
│   │   └── rm_serial_driver : 串口通信
│   │       ├── crc.hpp
│   │       ├── packet.hpp
│   │       └── rm_serial_driver.hpp
│   ├── launch
│   │   └── serial_driver.launch.py
│   ├── package.xml
│   └── src
│       └── rm_serial_driver.cpp
└── video_img_publisher : 供调试用的本地视频的图像流发布节点
    ├── CMakeLists.txt
```

```
├── config
├── include
│   └── video_img_publisher
│       └── video_img_publisher.hpp
├── launch
├── package.xml
└── src
    └── video_img_publisher.cpp
```

1.5.4.3 重要算法原理阐述

1. 二值化

对 cv::Mat 格式图像的所有像素点进行并行 forEach 操作，

如果己方颜色为红色：当且仅当该像素的 r 分量高于某阈值，g、b 分量低于某阈值，并且 r 分量与 b 分量的差值高于某阈值时，将输出图像对应像素置为 255（白），否则置为 0（黑）；

如果己方颜色为蓝色：当且仅当该像素的 b 分量高于某阈值，g、r 分量低于某阈值，并且 b 分量与 r 分量的差值高于某阈值时，将输出图像对应像素置为 255（白），否则置为 0（黑）。

阈值参数由 ros parameters 传入。

2. 图形筛选

locatePatterns()函数输入一个二值化后的图像，使用 OpenCV 的 findCoutours()获取其中的轮廓，而后对这些轮廓逐个进行初筛：

先假设该轮廓为 L 形，那么将轮廓单独绘制到一个作为画布的二值化图像上，使用 OpenCV 的 minEnclosingTraingle()函数找到轮廓的最小外接三角形，而后基于此三角形分别假设三条边的某一条边为 L 形角的对边，然后生成一个理想的 L 形模板，绘制到另一个二值化画布上，计算两个二值化画布的交并比（交集面积与并集面积之比，IoU），若 IoU 大于某个阈值则通过判定。为了避免小颗图块噪声，还需要增设一个交集面积最小阈值的判定条件。三次假设中若有一次判定通过，则认定该图块为 L 形图块。

如果不是 L 形，则假设该轮廓是兑换站上的方形图块，由于现阶段方案不考虑方形图块，该部分代码未经测验，暂不介绍。

如果不是方形图块，那么就舍弃该轮廓。

将每一个 L 形图块的最小外接三角形与其轮廓对象组成 std::pair，放入输出的 vector 容器内；将每一个方形图块的最小外界矩形与其轮廓对象组成 std::pair，放入另一个输出的 vector 容器内。每一个三角形对象中的三个点是有序排列的：从前到后为逆时针顺序，并且 L 形顶点的对应点须在第二位。

3. 构建最佳兑换站模型

buildModel()函数输入一个存储所有 L 形图块最小外接三角形的 vector 容器，以及一个存储所有方形图块最小外界矩形的 vector 容器。

判断两个 L 形图块为兑换站中相邻 L 形图块的指标为两条对应共线边的不对齐程度；关于两条分别以 p1, p2 和 p3, p4 为顶点的线段的不对齐程度的计算公式为：

(p3 到 p1p2 所在直线的垂直距离 + p4 到 p1p2 所在直线的垂直距离

+p1 到 p3p4 所在直线的垂直距离 + p2 到 p3p4 所在直线的垂直距离)

/ 四点最小外接圆的半径

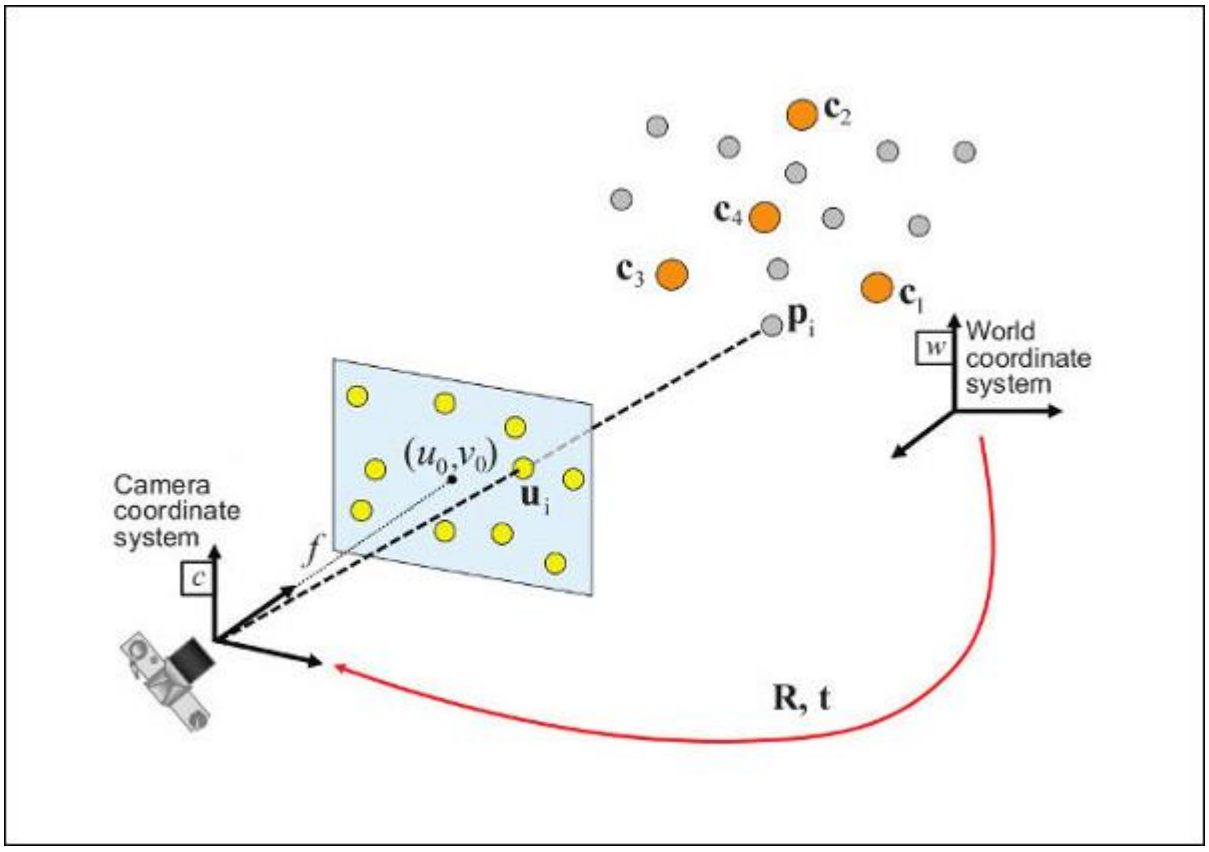
将三个 L 形图块按顺序组合即为一个兑换站模型，不考虑方形图块。将第一、第二个 L 形图块的不对齐度与第二、第三个 L 形图块的不对齐度相加作为总误差，使用剪枝 dfs 来从所有 L 形图块中寻找误差最低的组合模型。

检查并调换外接三角形，保证三个三角形的先后顺序仍为逆时针。先后取三个外接三角形的 9 个顶点，即为兑换站的 9 个参照点。

4. 解算姿态

solvePnP 和相关函数可以在给定一组物体点、给定它们在图像中对应的投影点，以及相机的内参矩阵和畸变系数的条件下估计物体的姿态，见下图。

(实际上相机系的 x 轴指向右侧，y 轴指向下方，z 轴指向前方)



使用相机透视投影模型 Π，和相机内参数矩阵 A（在文献中也记作 K），将世界系 X_w 中表示的点投影到图像平面[u,v]中:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{A}\,\mathbf{\Pi}\,{}^{\mathbf{c}}\mathbf{T}_{\mathbf{w}} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

因此旋转（rvec）和平移（tvec）向量即为所要估计的位姿。使用它们可以将世界系中表示的 3D 点转换到相机系中：

手动引用 https://docs.opencv.org/4.x/d5/d1f/calib3d_solvePnP.html

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = {}^{c}\mathbf{T}_{\mathbf{w}} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

因此，对图像进行上述分析处理可以得到 9 个兑换站参照点的投影点坐标，与相机内参，畸变系数，以及预设的兑换站系内 9 个参照点坐标等一起输入 solvePnP 函数中，而后便能解算出一个位置向量和一个旋转向量，即为 6D 位姿。将位姿从 opencv 相机系变换到本方案使用的相机系之后，即可发送给下位机。

1.5.4.4 算法评价与未来优化方向

由于兑换时底盘可动，识别时可以默认兑换站在水平方向上永远处在相机视野的正中间。本程序的目标是识别零级至三级位姿，即 x、z 坐标和 roll、pitch 角都有一定可变范围。

队内短焦相机的视场角为 70°左右，若要在相机 x、z 坐标不变的情况下使视野覆盖到所有可能的兑换站位姿，可能需要将相机以一定仰角安装。在调试得到最佳参数的条件下，目前的算法已经能够符合要求地识别视野内的零级至三级位姿，测试记录详见 1.6.1 节。

目前的算法思路仍未发挥全部潜能，未来还有若干优化方向：

- L 形最小外接三角形的不对齐度可以作为一个判断其属于所找兑换站模型的合格度指标，但是倘若两个 L 形满足共线边对齐的要求，但是一个远大于另一个，那么即使这两个图块明显不属于同一个兑换站模型，视觉算法也会认为它们两个的指标很好。一个可能的解决方法是对 L 形的内侧直角边也应用不对齐度检测，这样上述特例的内边不对齐，记录的总不对齐度会增加。
- 使用最小外接三角形定义 L 形图块的思路对仿射变换有很好的鲁棒性，但是相机中出现的投影变换是一个比仿射变换更大的集合，近大远小效应使得 L 形图块的内侧直角顶点与理想模型相比稍微偏内或偏外。在实际应用中这种偏差基本不影响识别概率和识别准度，但是如果要进一步优化识别概率的话，可以对理想模板中该顶点在一定范围内向内和向外微调， 遍历所有的微调采样点来找到最佳 IoU 和面积。
- 本算法未考虑第四个 L 形图块、两个方形图块、以及侧面的细 L 形图块。可以将前两者纳入分析范围以提高定位准度；如果相机能清晰识别侧面 L 形的 6 个角点，那么可以识别它使得算法有能力识别第四级兑换站 yaw90 度的情况。

1.5.4.5 算法接口介绍

本项目的 ros2 程序框架由 3 个节点组成，调整对应的启动参数可以改变程序的功能。

这里介绍 engineer_bringup.yaml 的各参数，这些参数会被转换成 ros2 parameter 赋给各节点。

```yaml
/video_img_publisher:
  ros__parameters:
    video_path: /xxxxx.mp4  # 调试用的本地视频路径
    fps: 30.0               # 视频发布帧率(Hz)
    fov: 30.0               # 视频所用相机视场角(deg)
    width: 3840             # 图像宽度(px)
    height: 2160            # 图像高度(px)

/monocular_img_publisher:
  ros__parameters:
    fps: 30.0               # 运行帧率
    fallbackFov: 30.0       # 若没有先前标定的内参文件
                            # 便通过该视场角参数以及图像宽高来估计相机内参(deg)
    exposureTime: 0.15      # 曝光时间(s)
    flip: False             # 翻转图像
    disableUndistort: False # 禁用畸变校正
    autoWB: True            # 自动白平衡

/exchange_posture_recognizer:
  ros__parameters:
    selfColor: red          # 己方颜色，可以为"red"/"blue"
    debugBinarize: False    # 是否开启debug二值化模式
                            # 该模式下仅对图像进行二值化并发布，而不进行后续处理
    runMode: publish        # 运行模式，可以为"publish"/"service"/"service-fake-posture"
                            # 分别对应发布模式，服务模式，服务+假位姿模式

    # 下述阈值中，rgb像素值取值于[0, 255]
    thresholdForBlue: [ 150.0, 255.0, 255.0 ] # 蓝色兑换站二值化阈值：minBlue, maxGreen, maxRed
    thresholdForRed: [ 50.0, 255.0, 255.0 ]   # 红色兑换站二值化阈值：minRed, maxGreen, maxBlue
    bgrSubtractForBlue: -1                    # 蓝色兑换站二值化阈值：max{Blue - Red}
    bgrSubtractForRed: 30                     # 红色兑换站二值化阈值：max{Red - Blue}

    minIoU: 0.65                              # 最小IoU阈值
    minLArea: 100                             # 最小交集面积阈值(px)
    Lparam: 0.18032786885245905               # 定义正面大L形图块粗细比例的参数
                                              # 具体定义为 L的臂宽 / 外接三角形直角边 = 11mm / 61mm

    fakePosture: [                            # 假位姿，齐次矩阵
      1., 0., 0., 300.,
      0., 1., 0., 50.,
      0., 0., 1., 200.,
      0., 0., 0., 1.,
    ]
```

1.6 研发迭代过程

1.6.1 测试记录

1.6.1.1 机械臂

可达性仿真测试

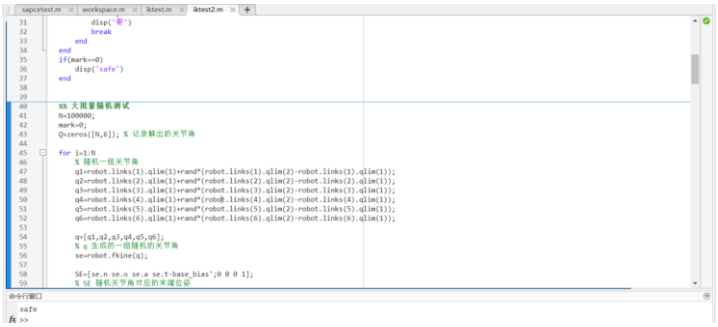
我们需要验证我们的机械臂是否一定能达到兑换站目标空间中的所有位姿（上面的工作空间只能看出位置，不能看出姿态），因此，我们需要进行兑换站所有位姿的可达性测试。

首先我们通过运动解耦的思想和一些解算顺序的策略，求出了这个构型的逆运动学解析解（详见 1.5.3.2 中的机械臂运动学方案设计）。由于这个工具箱本身具有正逆运动学求解的功能（逆运动学是迭代解），我们用此先来检验我们的解析解的正确性。

```
%% 单个随机测试
% 随机一组关节角
q1=robot.links(1).qlim(1)+rand*(robot.links(1).qlim(2)-robot.links(1).qlim(1));
q2=robot.links(2).qlim(1)+rand*(robot.links(2).qlim(2)-robot.links(2).qlim(1));
q3=robot.links(3).qlim(1)+rand*(robot.links(3).qlim(2)-robot.links(3).qlim(1));
q4=robot.links(4).qlim(1)+rand*(robot.links(4).qlim(2)-robot.links(4).qlim(1));
q5=robot.links(5).qlim(1)+rand*(robot.links(5).qlim(2)-robot.links(5).qlim(1));
q6=robot.links(6).qlim(1)+rand*(robot.links(6).qlim(2)-robot.links(6).qlim(1));

q=[q1,q2,q3,q4,q5,q6];
% q 生成的一组随机的关节角
se=robot.fkine(q);

SE=[se.n se.o se.a se.t-base_bias';0 0 0 1];
% SE 随机关节角对应的末端位姿

q2=ikk(SE,L,0);
% qq 用 SE 逆解出来的关节角，这个可能和 q 不一样，因为有多解

se2=robot.fkine(q2);
SE2=[se2.n se2.o se2.a se2.t-base_bias';0 0 0 1];
% SE2 用我们解的 q2 对应的末端位姿，只要这个和 SE 一样就说明逆运动学解的是对的
```

```
mark=0;
for ii=1:12
    if(SE(ii)-SE2(ii)>1e-4)
        mark=1;
        disp('寄')
        break
    end
end
if(mark==0)
    disp('safe')
end
```

以上是单个的测试，在通过 10w 随机数据的大批量测试后，我们认为我们的解析解公式是无误的。



解算测试结果

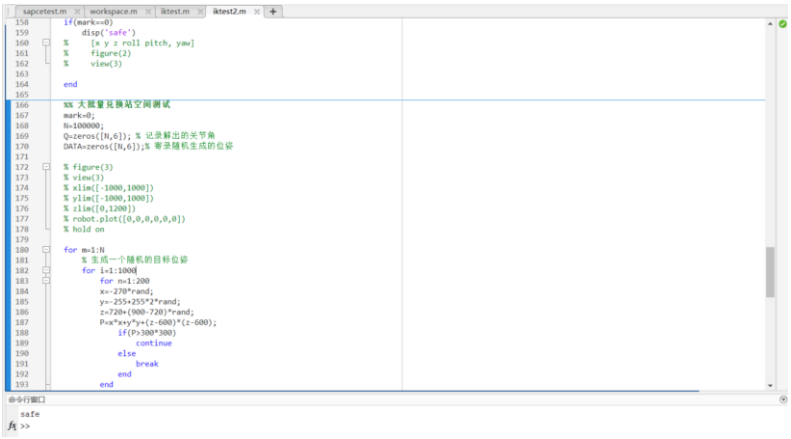
接下来，我们随机在官方规则给出的兑换站运动空间/角度范围内，生成一系列兑换站位姿，然后用我们的解算去解出相应关节角，通过大批量随机数据的检验，来确定我们的构型一定能满足规则的需求。

```
%% 单个随机兑换站空间测试
mark=0;

% 生成一个随机的目标位姿
for i=1:1000
    for n=1:200
    x=-270*rand;
    y=-255+255*2*rand;
    z=720+(900-720)*rand;
    P=x*x+y*y+(z-600)*(z-600);
```

```
        if(P>300*300)
            continue
        else
            break
        end
    end
end
x=-x;
y=-y;

pitch=(-60*rand)*pi/180;
roll=(-45+90*rand)*pi/180;
yaw=(-90+180*rand)*pi/180;

T=transl(x,y,z)*rpy2tr(-roll,pi/2-pitch,yaw); % 鬼知道官方的 rpy 旋转顺序是什么，规则沙龙上问了也没回我 QAQ
trplot(T,'length',200,'rgb');
hold on

SE=T;
SE(1:3,4)=SE(1:3,4)-base_bias'; % 减去底座的偏置，这个为目标位姿

q2=ikk(SE,L,1);
% qq 用 SE 逆解出来的关节角，这个可能和 q 不一样，因为有多解

robot.plot(q2);


se2=robot.fkine(q2);
SE2=[se2.n se2.o se2.a se2.t-base_bias';0 0 0 1]; % 这里也要减去偏置
% SE2 用我们解的 q2 对应的末端位姿，只要这个和 SE 一样就说明逆运动学解的是对的


for ii=1:12
    if(SE(ii)-SE2(ii)>1e-4)
        mark=1;
        disp('寄')
```

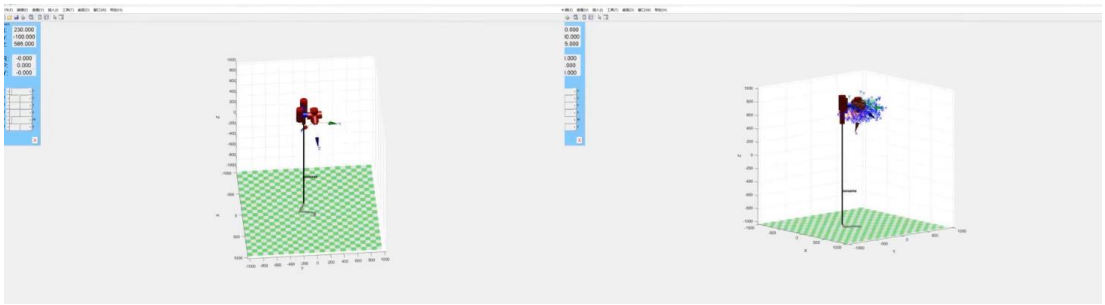
```
        break
    end
end
if(mark==0)
    disp('safe')
%     [x y z roll pitch, yaw]
%     figure(2)
%     view(3)

    end
```

以上是兑换站单个位姿的生成和解算测试，在通过 10w 随机数据的大批量测试后，我们认为我们的机械臂能够达到兑换站的所有目标位姿。



位姿到达测试结果

将这些随机数据产生的解利用可视化函数画出来，发现看起来页符合我们的要求，因此我们认为仿真的过程和结果没有问题。



产生的随机解（图截自完整形态视频，右图画出了每个解的末端的坐标系）

自定义控制器功能验证测试

一.测试前的准备

我们现有的兑换方案：

视觉：姿态很正基本上可以相信，位置有误差，没有防止碰撞的算法，鲁棒性比较差，只能用于兑换站和机械臂正契合的状态、

TODO：可以让位置很准，并写一套可能符合各种情况的自动化流程。

自定义控制器：已经完成所有功能，很安全。

TODO：Stuck 脏标，要在测试同时来判断什么时候 stuck。

现有的五套方案：

1.完全自定义控制器

2.自定义控制器异型控制+视觉

3.异型手动控制+视觉

4.手动控制

5.视觉加手动控制

二.测试结果可能判断

If（3，5 方案的快者比 4 方案平均快 5s 以上 || 2 方案比 1 方案快平均 5s 以上 || 操作手有想法）

{

视觉继续做；拉位置精度到差不多得了，换 A 板。

}

else

{

视觉搁置了，不用换 A 板

}

If（1，2 方案中的优秀者比 3，4，5 方案中的优秀者要好 || 操作手有想法）

{

自定义控制器为首要方案。

}

else

{

手动控制为首要方案。

}

PS：考虑到操作手生疏问题，先测自定义控制器方案。

我个人认为自定义控制器更优秀，所以摆在前面试。

三.测试方案设计

x，y 影响比较小，可以靠底盘解决，所以不做特殊处理，主要考虑 z，roll，pitch，yaw

表 5-14 不同难度等级的兑换槽位姿取值范围

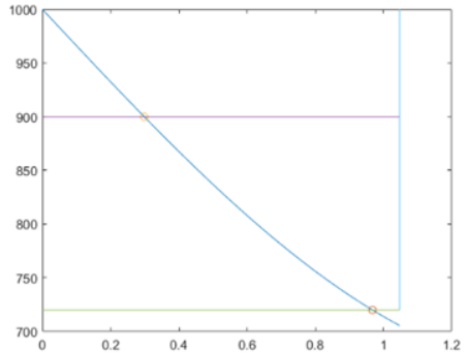
难度等级	x	y	z	θ	Φ	ψ
零级	-300	0	720	0	0	0
一级	-200	[-185,185]	720	0	0	0
二级	[-270, 0]	[-255, 255]	[720, 900]	0	0	0
三级	[-270, 0]	[-255, 255]	[720, 900]	[0，60]	[-45, 45]	0
四级	[-270, 0]	[-255, 255]	[720, 900]	[0，60]	[-45, 45]	[-90,90]

我们的臂的基本参数如下：

蓝色曲线为：1000（最高限高）- 340（臂末端+矿石）*sin（x）（pitch 角度）

左侧为能正常进去，右侧不能正常进去。

```
cross = 1×2
    0.9676  720.0000

cross = 1×2
    0.2985  900.0000
```



考虑到臂最大的 pitch 为 30°，在满足 30°的情况下高度分别选择 750 和 900 来区别两种情况。

（3 级+4 级混合测试）每次 10 组

3 级（45°roll，30°pitch，750z） （45°roll，30°pitch，900z）

4 级（45°roll，30°pitch，-60yaw，750z） （45°roll，30°pitch，-60yaw，900z）

（45°roll，30°pitch，60yaw，750z） （45°roll，30°pitch， 60yaw，900z）

暂定最多

6*2*5 =60 次

60/8 = 8 个气瓶

8/3 = 3 次气（1.30 个小时）

四.测试结果分析

	3级		4级	
兑换类型	45°roll，30°pitch，750z	45°roll，30°pitch，700z	45°roll，30°pitch，-60yaw，700z	45°roll，30°pitch，-60yaw，850z
完全自定义控制器				
自定义控制器异型控制+视觉	被操作手一票否决了，所以不用测试，他太喜欢自定义控制器了			
手动控制+视觉	42.56	50	35.93	40.39
		32.42	33.6	
		36.56	35.59	
手动控制	50.64	34.23	44.58	54.32
	41.35	31.72	45.84	47.9
	40.43	35.52	43.42	48.48
备注	手动时需要上抬一点塞进去	手动可以正常进去	手动可以进去，应当注意调节顺序，姿态-底盘-位移 而且把最后推一下作为标准流程是合理的	手动可以放一半蹭进去，那就是在前期调节的时候不动pitch，只动姿态其他两轴，放到1/3位置放进去 我朝，可以把矿滚进去

实验结果：（本次实验以操作手意见为主，所以未到实验计划次数就停止了测试）

结论 1：视觉有必要做

操作手意见，视觉要保留

有了视觉：对于高四级提升了 10s 左右，对于低四级提升了 10s 左右，对于三级提升不够明显

分析：三级的调试只需要调节 pitch，roll，对于高的三级，都需要把 pitch 拉的最高，蹭进去，所以有没有视觉没有差别，需要调节的只有 roll，仅需要调节 45°，所以视觉的优势不明显，而对于四级多了 yaw 的调节，所以效果明显。

结论 2：不需要异型控制

在操作手玩过一段时间后，实在无法驾驭异型控制，且操作手自信表示，手操可以达到同样的效果

结论 3：不需要自定义控制器和视觉混用

操作手表示：视觉和自定义控制器是相反的效果。如果两者同用就丧失了手部姿态一一对应的效果，且只要操作手歪一下，视觉识别的准确姿态就产生了误差，且操作手表示能用手很快做到视觉同样的效果。

结论 4：纯自定义控制器 > 视觉+手动 > 纯手动

操作手在尝试了很多次手动遥控后，还是觉得自定义控制器更丝滑顺手，可以同时调节 6 个自由度。

结论 5：四级的可能性很高

操作手需要测试，现在的情况是，在 pitch 和 z 不是很极限的情况下，30°，800mm 左右，操作手发现了能蹭进去的方法，并且尝试的几次，高四级比高三级慢 8s 左右。

对于能正常放进去的低三级和低四级，低四级比低三级慢 10s 左右，而在视觉的加持下，两者基本上没有差别。所以四级的可能性很高。

结论 6：一些基本数据

兑矿的自动化流程为 15s 左右，剩下的时间为操作手消耗的时间

	低三级	高三级	低四级	高四级
手动+视觉	34.49	42.56	35.04	40.39
纯手动	33.82	44.14	44.61	50.23

结论 7：兑换逻辑确定

纯自定义控制器 > 视觉+手动 > 纯手动

如果自定义控制器寄了换视觉+手动，如果自定义控制器和视觉都寄了，

上手动和虚拟三轴【操作手觉得自动化拿矿出来的时间（现在是 10s，感觉最高减到 5s）可以用来对姿态，画 UI 的虚拟三轴，自动的时间，操作手就可以操作，虚拟三轴给予操作手反馈，然后等自动化结束直接映射到末端三轴上】

结论 8：理论计算有点不可信

我们的末端有角度误差，车体有误差，视觉识别有误差，纯靠视觉识别的位置姿态来理论计算能不能正着进去，需不需要蹭进去不可取，容易给操作手误判。

结论 9：兑换之后的工作（按优先级排序）

1.确定一个新的好一点的图传角度（拆图传测试）

2.视觉测试，保证各个位置的姿态识别准确，位置就别管了，保姿态就行。

3.换 A 板，整合这两周写的代码+刚刚说的新功能

4.拉快自动化流程（等新车）

5.补全 UI

补充测试（自定义控制器）

	3级		4级	
兑换类型	45°roll，30°pitch，750z	45°roll，30°pitch，700z	45°roll，30°pitch，-60yaw，700z	45°roll，30°pitch，-60yaw，850z
完全自定义控制器	26.7	21.12	23	50
	38	23.76	34	38
	23	22.35		
自定义控制器异型控制+视觉			34	32
			22	33
			20	
手动控制+视觉	42.56	50	35.93	40.39
		32.42	33.6	
		36.56	35.59	
手动控制	50.64	34.23	44.58	54.32
	41.35	31.72	45.84	47.9
	40.43	35.52	43.42	48.48
备注	手动时需要上抬一点塞进去	手动可以正常进去	手动可以进去，应当注意调节顺序，姿态-底盘-位移 而且把最后推一下作为标准流程是合理的	手动可以放一半蹭进去，那就是在前期调节的时候不动pitch，只动姿态其他两轴，放到1/3位置放进去 我朝，可以把矿滚进去

结论 10：自定义控制器再在三级状况下明显由于视觉+手动，快将近 10S 左右，对于 4 级也微小胜过视觉+手动，所以自定义控制器的优先级优于自定义控制器。

结论 11：自定义控制器+视觉虽然并未展现出很好的优势，但其保持了很好的稳定性，据操作手“交代“过大的 yaw 会导致手势反人类，扭曲的手腕可能会导致兑换的问题，而有了视觉的辅助，可以保证操作手的手始终保持在一个合适的位置。

复活赛前整体功能指标测试

储矿测试

取矿方式：单矿模式，三连取模式

测试道具：工程车，1-5 号矿石

测试目标：测试并记录不同程度歪的矿石储矿时可能会发生哪些问题，发现所有的错误情况；测试理论正常情况的储矿成功率

上场目标：

正常储矿成功率 100%

测试流程：

在单矿模式下，测试员手持矿石，以一些比较歪的角度喂给工程车，测试并记录不同程度歪的矿石储矿时可能会发生哪些问题

在三连取模式下，测试员手持矿石，以一些比较歪的角度喂给工程车，测试并记录不同程度歪的矿石

储矿时可能会发生哪些问题，前面单矿模式的所有错误情况都要测试

测试记录：

切左边角和底边角（正角度，最大左测角，最大右测角）

切右边角和底边角（正角度，最大左测角，最大右测角）

每种情况测试 3 次以上，储矿成功率 100%。初次之外的普适情况测试了 10 次左右。

取矿&储矿测试

取矿方式：单矿模式，三连取模式

测试道具：工程车，资源岛，1-5 号矿石

测试目标：测试不同取矿模式的成功率，可随测试调整参数

上场目标：

单矿模式成功率 100%

三连可调成功率 100%

~~三连自动取成功率 80%~~（这个模式要联系，练到 80%为止，10 次感觉不太够）

储矿成功率 100%

熟悉各种取矿的组合（1+2，2+1）

测试流程：

操作手操作工程车，在小资源岛获得三个矿石，每个模式测试 10 次，统计成功率

在整个测试过程中，在矿石角度不太离谱（操作手失误）的情况下，记录拿起来的矿石储矿的成功率

根据测试中出现的问题，可以随时调整

测试记录：

取矿方式	1	2	3	4	5	6	7	8	9	10	AVG	
3+0	40''	35''	34''	39'' 掉第一个矿	35''	35''	36''	36''	36''	35''	36.1	第一个矿左右不需要对太正，利用第一个矿的位置进行底盘修正，稍后的两个矿就很正常 需要ui对位中间矿 第二个矿会卡后挡板，可以自动化解决
2（III）+1（I）	50''	46''	52''	38''	38''	77'' 操作手wifi离线 一矿卡矿舱，二三矿对位失败导致取两次	37''	38''	37''	41''	45.4	时刻记得检查G的情况 ~~一矿模式也需要加ui~~
1（I）+2（III）												取消，相当于2+1倒叙

总结：取三个矿的时间平均为 36.1s，取 2+1 的矿由于操作原因比纯 3 矿复杂平均时间为 45.4s，时间差距 9.3s。可以靠联系弥补，2+1 的操作时长最短稳定在 37-38s，自动化差距时间和纯 3 矿估计在 1-2s.

兑换测试

兑换模式：视觉+自定义控制器，自定义控制器，视觉+手动，手动，**自定义控制器中途切换手动**

测试道具：工程车，兑换站，1-5 号矿石

上场目标：

不同东西挂掉的逻辑切换没有问题，绝对不会疯

测试流程：

在不同的兑换模式下，测试若干组兑换流程（3 矿连续兑换），记录有无出现程序逻辑问题

每种模式至少保证 10 组（3 连）以上的测试量

记录操作手开车的情况下视觉正常触发的概率

测试记录：

兑换测试：

level	index	x	y	z	yaw	pitch	roll	Time 视觉+自定义控制器	Time 自定义控制器	Time 手动	Time 自定义控制器中途切换手动	记录 自定义控制器	记录 手动	记录 视觉+自定义控制器
3	1	-140	235	720	0	-30	-35	null/null/23"	18"	36"			需要考虑一下姿态三轴与位置三轴的控制按钮联动情况	第一次 视觉灯闪烁，识别失败，可能离太远了，建议ui加视觉可视空间 第二次 遥控器连接掉了，建议加个锁扣之类的方案，避免正式比赛出现这种问题
3	2	-140	235	720	0	-30	35		32"			动捕点		
3	3	-170	156	744	0	-40	-25	24"	22"	28"		ui卡了		动太快
3	4	-170	156	744	0	-40	25		20"61					
3	5	-200	78	768	0	-45	15	null/28"	24"	43"			需要考虑左摇杆的三控制顺序	第一次 没录到时间，sbslj 第二次 不应该动底盘的，应该在差不多的位置直接动臂，依靠ui判断臂运动空间
3	6	-200	78	768	0	-45	-15		37"					
3	7	-230	0	792	0	-50	5	20"	27"	fail/28"		e后如果未碰过了底线，再按q，大概率卡储矿；需要vb复位 vb不会断气，进q才会	第一次 能做到，但是必须让矿自由掉落下去，所以需要找清楚位置 考虑需要对超高情况做提示 第二次 完美找到卡掉矿的位置，但是仍然需要考虑找到ui提示的方法（矿石是否进入一半？是否超高？）	
3	8	-230	0	792	0	-50	-5		21"					

4	1	-270	0	725	90	-40	-45		50"/25.5"	38"		第一次测试时打到调节钮，导致全部电机死了		
4	2	-235	135	725	65	-30	-40		null/14"2/32.6" --- 上述所有数据测量距离有问题 --- null/21"	55"		第一次存在越界情况 第二次矿石rfid识别有问题，需要再抖一下兑换站 第三次 纯rfid识别有问题 --- 上述所有数据测量距离有问题 ---	好尼玛难搞啊	
4	3	-270	45	720	90	-40	-45		40"/null/28" --- 上述所有数据测量距离有问题 --- 20"			第一次有点难度，加越界加必须瞄进去 第二次 机械臂抽搐加代码死掉了，初步怀疑是线缆动或者视觉越界，排障为越界检测代码冲突 第三次 极限进入 --- 上述所有数据测量距离有问题 --- 臂把兑换站打下去了40（		
4	4	-250	85	735	65	-45	-35		fail/fail/52" --- 上述所有数据测量距离有问题 ---	35"		第一次 矿掉了，操作问题导致自定义控制器乱 第二次 根本进不去 第三次 矿石进入的逻辑不是放入完全，而是只要有一个边进去一半及以上，那么就有机会 --- 上述所有数据测量距离有问题 ---	可以把矿扔进去	
~~4~~	~~5~~	~~-270~~	~~0~~	~~720~~	~~90~~	~~-40~~	~~-45~~							

4	6	-270	0	720	90	-45	-30		fail --- 上述所有数据测量距离有问题 --- 17"	52"		无法兑换 --- 上述所有数据测量距离有问题 ---		
~~4~~	~~7~~	~~-270~~	~~35~~	~~720~~	~~90~~	~~-40~~	~~-35~~							
4	8	-270	35	720	90	-60	-20		fail --- 上述所有数据测量距离有问题 --- 17.5"	fail/42"		无法兑换 --- 上述所有数据测量距离有问题 ---	第一次 想扔进去，但是前进的距离太小，导致矿卡住了 建议要么多深入一点，要么就顶撞3兑换站 第二次 需要考虑一下如果兑换一些奇怪角度，有可能在运动中卡到兑换站，最坏的情况是爆掉臂	

补充测试：视觉+自定义

		x	y	z	yaw	pitch	roll	Time 视觉+自定义控制器	记录
?	?	-150	0	750	90	-45	-45	fail	see不了一点
?	?	?	?	750	60	-45	-45	50″/28″	第一次 按上去兑换站了，有点怪 第三次 realsense的坐标转换是基于其本身的坐标系，而不是手的坐标系，需要在操作的时候有所注意 以及在控制的时候发现姿态的对应也会比较难受（抬手），决定拉大映射到2倍
?	?	?	?	750	30	-45	-45		
?	?	?	?	?	?	?	?	20″ 36″ 19″ 22″ 18″ 23″ 30″ 20″ null 17″	需要磨合视觉，所以加入了以下几条测试，并加上时间记录 第一次 需要注意有无过线，第一次没过 第二次 self-state闪了一下 第四次 注意roll的方向 第五次 还是需要注意roll的方向 第七次 视觉寄了 第八次 视觉寄了，频闪，怀疑是反光 第九次 视觉寄了，而且未注意方向 第十次 视觉寄了，打到兑换站一次，不清楚啥情况。

结论：

三级：

测试方式	测试次数	最小值	平均值	成功率	结论
手动	4+1（fail）	28	33.75	80%	手动还是和自定义控制有差距，且由于放气后的调整蹭矿比较难，矿可能会掉，有对不上的可能
自定义控制器	8	18	25.125	100%	
自定义控制器+视觉	4+3（未记录数据）	20	23.75	100%/100%	视觉对自定义控制器的加成是有的，三级的自定义控制器识别准确性暂时是 100%，虽然最小值有视觉后变小，估计是操作还没有熟悉

四级：

测试方式	测试次数	最小值	平均值	成功率	结论

手动	4+1（fail）	28	33.75	80%	手动还是和自定义控制有差距，且由于放气后的调整蹭矿比较难，矿可能会掉，有对不上的可能
自定义控制器	8	18	25.125	100%	
自定义控制器+视觉	4+3（未记录数据）	20	23.75	100%/100%	视觉对自定义控制器的加成是有的，三级的自定义控制器识别准确性暂时是 100%，虽然最小值有视觉后变小，估计是操作还没有熟悉

自定义控制器纯吃熟练度，后面的 4 级测试效果甚至比 3 级好，有自定义控制器，三级四级差不多。

有视觉有 1-2s 的提高，但是需要磨合，突然上会变操作手会变唐。（坏消息是视觉的成功率并不稳定，好消息是有提高，坏消息是提高只有 1-2s，没了影响不大）

对于解出来裁判系统的数据最极限的位置为

-270	45	720	90	-40	-45

并没有出现够不到的情况。但估计还是会有够不到的情况。

1.6.1.2 气动系统

真空度相关测试

减压阀型号：AR20-02E-B（车上） /AR-2000（备用）

吸盘型号：zptH80

新真空发生器型号：zh07b

测试 0：真空气路密封性测试

测试目标：把阀岛到吸盘前的气路调到不漏气

测试结果：**是**/否密封

测试 1：工作真空度

目标：确定上场的工作真空度；记录下短气路情况下的真空气流量

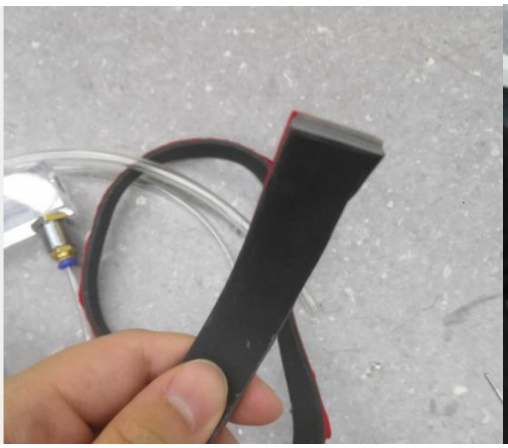
测试工具：空压机，zh07b 真空发生器，中等消耗程度的矿石 x1

检测标准：连续吸 7s，不能有吸盘印记

视实际情况增加【不同真空度】和【吸盘处理情况】的情况

工作真空度（-kpa）	工作气压（Mpa）	吸盘处理情况	矿石表面情况
40	0.32	无	R 标处无印记，空白处有印记
50	0.35	无	R 标处无印记，空白处有印记（**后续测试只测 R 标处**）
60	0.36	无	有印记
60	0.36	橡胶垫	很浅的印记
70	0.38	橡胶垫	很浅的印记
70	0.38	海绵垫	无印记
80	0.4	橡胶垫	很浅的印记
80	0.4	海绵垫	无印记
90	0.5	海绵垫	有印记

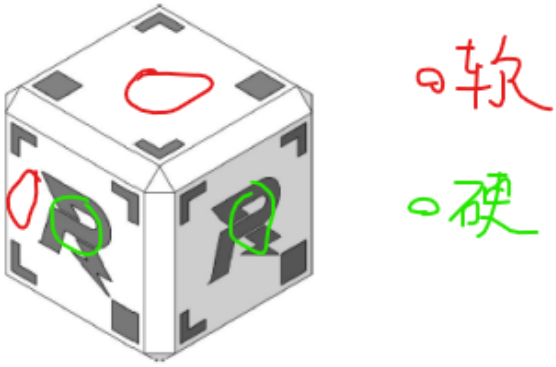
橡胶条示例：



海绵条示例：

结论：

- 有印花处和无印花处的硬度不一样，如图，有印花（如 R 标）处较硬，不容易出印子
  软的地方 40mpa 就会出印子，搞不了一点



- H80 的吸盘使用前，用 5mm 海绵胶把里面贴住
- 上场工作真空度为 70-80 ，此时工作输出气压为 0.38-0.4 （非车上气路，仅参考用）
- 短气路情况下，吸附时的真空气流量为 3

测试 2：工作气压

目标：在实车的气路上确定【工作气压】；记录下车上气路情况下的真空气流量

测试工具：高压气瓶，车上气路，中等消耗程度的矿石 x1

测试流程：东西都接到车上，确定使末端真空度达到目标真空度的实际工作气压

测试结果：工作气压为 0.4-0.45（建议 0.45）；车上气路情况下的真空气流量为 1.5

末端真空度（-kpa）	工作气压（mpa）
40	0.33
60	0.4
70	0.42
80	0.45
85（极限）	0.5

测试 3：工作时间

目标：验证【工作时间】是否达标，末端真空度在 -80 kpa 下持续工作 6min 即为成功

测试结果：是/否成功

17mpa 的气瓶，持续工作 7min

1.6.1.3 识别兑换站位姿

识别算法准度测试

测试的基本思路是，事先在实验室的桌上划定坐标轴（坐标系为 z 朝上，x 朝相机右侧，y 朝相机后侧），测试时固定相机和兑换站模型，测量相机镜头正表面中心的坐标；通过将 STM32 开发板放置于兑换站表面，使用其陀螺仪测量兑换站的实际位姿，再通过测量兑换站右下角的坐标，经程序计算出兑换站在相机系内的实际位姿，最后将其与识别算法识别得的位姿比对，并观察误差。

由于识别算法实际得到的是兑换站相对于相机光心的坐标，而非相对于镜头正表面中心的坐标，那么

识别到的所有坐标会有一个 x 坐标的一个固有偏差，理论上可以在测试之后取 x 坐标误差平均值，在算法中增加一个坐标偏置来修正它。

在实际操作中相机的 yaw 朝向经常无法被精确调节。因此，在真正测试之前需要先将兑换站摆到中间位置，使得期望的 y 轴数据为 0，进行一次识别，而后观察识别数据，微调相机将 y 坐标调至 0mm 左右，从而校准相机 yaw 轴。这种做法严格意义上会使得测得的数据并非实际数据，因为校准 yaw 轴时识别算法的误差无法排除；但如果这种情况下测得的所有数据误差都相对均匀地分布在足够小的范围内，那么依然可以证明误差满足要求。

下图中使用的表格长度单位皆为 mm。

平移准度测试

备注	相机-世界系坐标			兑换站右下角-世界系坐标			测得 兑换站姿态四元数				计算得 兑换站中心-相机系坐标			识别成功与否	识别得 兑换站中心-相机系坐标			识别得 兑换站姿态四元数				兑换站中心-相机系坐标 error			识别得 兑换站姿态四元数 error			
	x	y	z	x	y	z	x	y	z	w	x	y	z		x	y	z	x	y	z	w	x	y	z	x	y	z	w
3.27 x距离500 近中心 右极限																												
校准相机yaw，但是z误差很选异	0.0	400.0	220.0	-500.0	-253.5	0.0	0.000	0.000	0.000	1.000	-502.0	0.0	-72.5	成功	-512.2	-0.8	-68.91	-0.010	-0.005	0.009	1.000	-10.22	-0.80	3.59	-0.01010	-0.00456	0.00922	-0.00010
x=500 右极限	0.0	400.0	220.0	-500.0	0	0.0	0.000	0.000	0.000	1.000	-502.0	253.5	-72.5	50%成功	-518.8	224.2	-73.83	-0.010	-0.008	0.023	1.000	-16.82	-29.25	-1.33	-0.01019	-0.00770	0.02289	-0.00034

3.27 日计划测试兑换站在视野中心以及上、下、左、右四个方向上的极限，但是在测试右极限的时候发现 y 坐标的误差达到了惊人的 3cm；此时在测试画面中可见相机的畸变很严重，因此我们重新标定了该相机的内参。

备注	相机-世界系坐标			兑换站右下角-世界系坐标			测得 兑换站姿态四元数				计算得 兑换站中心-相机系坐标			识别成功与否	识别得 兑换站中心-相机系坐标			识别得 兑换站姿态四元数				兑换站中心-相机系坐标 error			识别得 兑换站姿态四元数 error			
	x	y	z	x	y	z	x	y	z	w	x	y	z		x	y	z	x	y	z	w	x	y	z	x	y	z	w
新内参校准相机yaw	0.0	400.0	220.0	-500.0	-253.5	0.0	0.000	0.000	0.000	1.000	-502.0	0.0	-72.5	成功	-525.9	-0.3	-70.82	-0.011	0.016	0.061	1.000	-23.92	-0.30	1.68	-0.01149	0.01598	0.06130	-0.00021
旧内参校准相机yaw	0.0	400.0	220.0	-500.0	-253.5	0.0	0.000	0.000	0.000	1.000	-502.0	0.0	-72.5	成功	-513.5	-0.0	-70.47	-0.011	-0.023	0.009	1.000	-11.47	-0.01	2.03	-0.01141	-0.02302	0.00893	-0.00037
新内参 x=500 右极限	0.0	400.0	220.0	-500.0	-50	0.0	0.000	0.000	0.000	1.000	-502.0	203.5	-72.5	成功	-523.7	202.3	-72.67	-0.008	-0.001	-0.001	1.000	-21.75	-1.21	-0.17	-0.00792	-0.00066	-0.00103	-0.00003
旧内参 x=500 右极限	0.0	400.0	220.0	-500.0	-50	0.0	0.000	0.000	0.000	1.000	-502.0	203.5	-72.5	成功	-511.3	202.6	-73.24	-0.009	-0.011	-0.015	1.000	-9.27	-0.90	-0.74	-0.00949	-0.01110	-0.01499	-0.00022

但标定后重测的数据显示，标定前后的数据质量并没有明显区别，而且之前的 3cm 误差也都降成至多 3mm 以内了。同时在测试的过程中我们发现相机的右极限变小了，也即相机的视野变小了。排除了各种原因后，我们怀疑这种现象是标定时进行的一次对焦操作导致的，由此可以推测相机对焦准度可能对测量准度有很大影响。在之后的测试中我们再未调整过相机的对焦。

备注	相机-世界系坐标			兑换站右下角-世界系坐标			测得 兑换站姿态四元数				计算得 兑换站中心-相机系坐标			识别成功与否	识别得 兑换站中心-相机系坐标			识别得 兑换站姿态四元数				兑换站中心-相机系坐标 error			识别得 兑换站姿态四元数 error			
	x	y	z	x	y	z	x	y	z	w	x	y	z		x	y	z	x	y	z	w	x	y	z	x	y	z	w
校准相机yaw	0.0	400.0	220.0	-500.0	-253.5	0.0	0.000	0.000	0.000	1.000	-502.0	0.0	-72.5	成功	-527.9	0.3	-70.35	-0.011	-0.021	-0.012	1.000	-25.94	0.30	2.15	-0.01130	-0.02100	-0.01205	-0.00036
x=500 右极限 delta y=250	0.0	400.0	220.0	-500.0	-50.0	0.0	0.000	0.000	0.000	1.000	-502.0	203.5	-72.5	成功	-528.0	201.0	-72.74	-0.011	-0.012	-0.001	1.000	-25.96	-2.55	-0.24	-0.01056	-0.01206	-0.00057	-0.00013
x=500 左极限 delta y=250	0.0	400.0	220.0	-500.0	550.0	0.0	0.000	0.000	0.000	1.000	-502.0	-196.5	-72.5	成功	-529.1	-197.7	-66.48	-0.008	-0.020	-0.017	1.000	-27.13	-1.15	6.02	-0.00767	-0.01952	-0.01732	-0.00037
x=700 下极限 delta z=420 y手动校准	0.0	400.0	420.0	-700.0	-253.5	0.0	0.000	0.000	0.000	1.000	-702.0	0.0	-272.5	成功	-729.9	-0.6	-265.76	-0.012	-0.011	0.000	1.000	-27.87	-0.61	6.74	-0.01223	-0.01097	0.00015	-0.00013
(3.29补) 上极限 deltaz=300	0.0	300.0	15.5	-700.0	-153.5	173.5	0.000	0.000	0.000	1.000	-702.0	0.0	305.5	成功	-721.5	-0.4	294.18	-0.009	-0.009	-0.008	1.000	-19.55	-0.40	-11.32	-0.00861	-0.00900	-0.00752	-0.00011

3.28 日我们测试了算法在四个视野极限中的表现，结果显示识别的位置坐标误差皆在 1.2cm 以内，除了上极限之外的其余数据误差都在 7mm 以内，能够满足识别要求。

事实上表中的测得兑换站四元数皆为未经测量的估计值，实际操作时将兑换站模型置于桌面上时兑换

站会有一个小角度的 yaw 上仰。但是从数据中来看，该偏差并未造成位置坐标的较大误差。

平移+朝向准度测试

3.29 日更换了新坐标系为：z 朝上，x 朝相机左侧，y 朝相机前方，右手系。

备注	相机-世界系坐标			兑换站右下角-世界系坐标			测得 兑换站姿态四元数				计算得 兑换站中心-相机系坐标			识别成功与否	识别得 兑换站中心-相机系坐标			识别得 兑换站姿态四元数				兑换站中心-相机系坐标 error			兑换站姿态四元数 error				兑换站姿态四元数 error（朝向单位矢，转角角度）			
	x	y	z	x	y	z	x	y	z	w	x	y	z		x	y	z	x	y	z	w	x	y	z	x	y	z	w	x	y	z	w
校准相机yaw	0.0	300.0	220.0	433.0	83.0	123.0	0.005711	0.010112	0.047180	0.998862	430.4	28.4	51.6	成功	448.5	28.6	50.1	0.000034	-0.013832	-0.015161	0.999789	18.09	0.18	-1.54	0.005677	0.024144	0.062341	0.000927	0.005678	0.024157	1.727326	3.113123
上+仰极限 roll0	0.0	300.0	220.0	388.0	105.0	273.0	0.003630	0.225129	-0.009363	0.974273	457.4	49.7	185.1	成功	483.8	41.3	183.44	0.012843	0.241008	-0.033549	0.969858	26.41	-8.42	-1.65	0.009213	0.015879	0.024186	0.004415	-0.009508	-0.016005	0.096136	-2.157024

本次测量将兑换站以高 pitch 角置于视野的上部，初探了相机的 pitch 上极限，经过计算，第二条数据中兑换站在相机视野中兑换站所占的角度为 25 度，若尝试将兑换站 pitch 角调得更高，即使兑换站关键图块没有越出视野，也会识别失败；因此我们推测该算法可能存在一个最低覆盖像角的极限。

同时误差结果显示，对朝向的解算中，非零旋转的转轴单位向量的误差在 10%以内，转角的误差在 4°以内。

基于此我们使用 Geogebra 做了估计，发现若最低覆盖像角极限存在，那么相机安装位置的可选范围将大幅收窄，而且相机需要仰视安装。

3.31 日，在测试新样本的过程中，通过观察程序的调试输出我们进一步优化了算法的两个阈值参数：minIoU 参数，以及 minLArea 参数。

在我们使用的相机(1920x1200)中，4 个 L 形图块的面积普遍超过 3 位数的像素，其他图块的面积普遍不超过 3 位数，因此将 minLArea 阈值设定为 100；前三个 L 形图块的 IoU 普遍超过 0.7，第四个 L 形图块从未超过 0.65，因此将 minIoU 阈值设定为 0.7。

调整参数后，从第三次数据可以看出对极端情况的识别概率有了很大的提升，第三次数据的纵向覆盖视角缩到了 21°，同时实验中偶发的误识别现象也完全消失。在兑换站上位、高 pitch 角、roll 角 45°放置的极限上，位置误差升到了 1.2cm 左右，但四元数朝向数据中转轴单位向量的误差仍然在 10%以内，转角的误差仍在 4°以内，因此这样的误差不应超出兑换站的物理容错范围。

同时注意到，此次极限测试中若提升 pitch 角，相机图像内最下方的红色 L 形图块会异常地从下方缩小，因此可以推断这里的 pitch 极限已不是由算法能力导致，而是由于兑换站模型在设计上框架四角凸出，遮挡对应灯条导致。即使仍取该次数据 21°为最低覆盖像角，能给出的可安装范围依然不小。

由此可以初步得出结论：兑换站识别算法能够满足所有三级位姿的准确识别需求。

备注	相机-世界系坐标			兑换站右下角-世界系坐标			测得 兑换站姿态四元数				计算得 兑换站中心-相机系坐标			识别成功与否	识别得 兑换站中心-相机系坐标			识别得 兑换站姿态四元数				兑换站中心-相机系坐标 error			兑换站姿态四元数 error				兑换站姿态四元数 error（朝向单位矢，转角角度）			
	x	y	z	x	y	z	x	y	z	w	x	y	z		x	y	z	x	y	z	w	x	y	z	x	y	z	w	x	y	z	w
校准相机yaw	0.0	300.0	214.9	480.0	134.0	107.0	-0.027280	-0.033826	0.004805	0.999044	470.9	-11.7	32.0	成功	501.6	-11.6	29.6	0.018886	-0.016863	-0.035216	0.999111	30.75	0.08	-2.43	0.011394	0.016963	0.040021	0.000067	-0.011407	-0.016953	0.943256	0.178968
中+平视 Roll45 重新校准yaw（	0.0	300.0	214.9	454.0	252.0	62.0	0.403461	0.034924	0.065043	0.912014	459.9	-58.0	55.6	成功	483.4	-58.3	53.5	0.406148	0.068539	0.031457	0.910690	23.56	-0.30	-2.08	0.002687	0.033615	0.033586	0.001324	-0.003398	-0.033575	0.082428	-0.368521
上+仰极限 Roll45（视角21度）	0.0	300.0	214.9	455.0	334.0	363.0	0.361166	0.368658	-0.248255	0.819768	617.6	9.6	276.3	成功	649.7	-2.6	262.98	0.377203	0.397857	-0.242300	0.809449	32.15	-12.18	-13.31	0.016037	0.029199	0.009953	0.019319	-0.022612	-0.029505	-0.029248	-3.777215

我们后续依然会持续测试更多的样本数据，以更完整地验证这一结论。

1.6.2 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（工程机器人 v0）	功能： 实现设想的各种控制模式，能够手动操控或追踪空间轨迹； 实现 0-3 级兑换，以及部分情况的 4 级兑换。	2023.3.15
工程机器人 V1.0	功能： 兑换 0-2 级矿石，以及部分 3 级情况的兑换； 储存 4 个矿石（不够稳定）； 全向移动。	2023.4.10
工程机器人 V1.1	功能： 兑换 0-2 级矿石，以及部分 3 级情况的兑换； 稳定储存 2 个矿石； 全向移动。	2023.5.2
工程机器人 V1.2（分区赛版本）	功能： 改良末端执行器，稳定取银矿，一次一个； 兑换 0-2 级矿石，以及部分 4 级情况的兑换； 稳定储存 2 个矿石； 全向移动。	2023.5.20
工程机器人 V2.0（国赛版本）	功能： 末端三轴重构，稳定三连取银矿； 稳定兑换 3-4 级矿石； 稳定储存 2 个矿石； 全向移动。	2023.7.23

版本号或阶段	功能或性能详细说明	完成时间

1.6.3 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	机械臂 4 级兑换有部分位姿不能到达	加上 pitch 角度后在不超过限高的情况下能兑换的矿石高度有限	在 pitch 角大于 35°时，可以更改兑换时的矿石持有方向	工程机器人 v1.0	机械工程师：李崇珊 嵌入式软件工程师：盛李杰
2	从侧面不能获取矿石	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	从上方取出矿石	工程机器人 v1.0	机械工程师：李崇珊 嵌入式软件工程师：盛李杰
3	储矿时矿石卡住	上层轮组和侧面轮组间距过小；上层和下层间距过小，且无挡板	调整间距；增加挡板	工程机器人 v1.0	机械工程师：刘乐
4	救援时左右移动容易脱钩	被救援杆的间距过大，救援钩爪过长	减小这两个尺寸到合适的程度	工程机器人 v1.0	机械工程师：张宸翰
5	大吸盘吸矿力不够	应该是真空发生器管子没插紧	先排查了阀岛-真空发生器，无漏气 换了个真空发生器解决，返回原来的真空发生器也解决 省流：拔插真空发生器解决	工程机器人 v1.2	机械工程师：李崇珊
6	小吸盘失效	末端修改时做了想当然的决策，用了没有测试的数据末端造出来之后也没有测试	排查了气路密闭性和气源输入压，排查了所有元件 测量末端真空度后发现无问题，结论是挡板太高，导致真空吸盘吸不到第二层，导致失效	工程机器人 v1.2	机械工程师：李崇珊
7	末端限位失效	设计问题，该弯的东西就会弯	掰回去，但精度寄；有待下一版改善	工程机器人 v1.2	机械工程师：李崇珊
8	吸盘不开	阀型号有问题，针太短插不紧	阀岛接触不良，拆了后壳解决	工程机器人 v1.2	嵌入式软件工程师：盛李杰
9	取储矿失败	复位位置影响储矿精度，太快了对复位精度要求太高了把吸矿的持续时间变长了	重新复位一次解决了	工程机器人 v1.2	嵌入式软件工程师：盛李杰

10	pitch 轴测试过程中掉电	新版走线的方式导致分电板处线材过多过硬，容易把电源线拉松，松一点也会有问题，但只有 pitch 出问题，同等情况下，roll 未出现问题，怀疑是分电板接口和 pitch 延长电源线做的有问题。	拉扯处增加预紧力，电源口打更多胶，从做 pitch 的线	工程机器人 v2.0	嵌入式软件工程师： 盛李杰
11	该亮灯的下一位也可能会亮，影响还挺严重的，光耦板疯了	线材固定点为焊接处导致的信号不稳 气路修改成某组气缸的开闭不在相邻的两路（抬升和救援交换一条线），因为最多错到下一位，错开接可以避免这个问题（1357…2468…这样接）	代码增加一些容错，应该不会再出问题了。但 bugg 有点难复现，三年用阀岛第一次出这个情况，希望改硬软件后不会再有，等待压力测试，机械错着插线	工程机器人 v2.0	嵌入式软件工程师： 盛李杰
12	30 弹速 17mm 小弹丸会打坏鸡笼网保护	单层鸡笼网强度不够，同时被击打时局部受力过大	保护使用双层鸡笼网，之间使用海绵减震	工程机器人 v2.0	机械工程师：张宸翰
13	17mm 弹丸会从车侧面进入，卡入左后轮，进入内部	侧面处保护上方有空隙	使用扎带与布吉胶将缝隙封闭	工程机器人 v2.0	机械工程师：张宸翰
14	气瓶架门上下端受击会有向内较大形变	门上下端是空的，没有防形变的限位	在气瓶架上下端增加 3D 打印限位	工程机器人 v2.0	机械工程师：张宸翰
15	轮组电机减速箱松动 在长时间抵墙测试后松动	减速箱居然会松 官方默认应该没打胶	拆下来打胶	工程机器人 v2.0	机械工程师：李崇珊

16	阀岛受到撞击会莫名其妙的掉电	工程车是导电的，碳纤维底盘铝方管连接了所有的车体，同时怀疑电气有破损连接了车壳，如果电压有波动（轮子转会导致，臂动不会导致）同时瞬间将车壳和某个导体连接（视觉墙的锁扣）就会放电导致电压降低，电磁阀对这种电压比较敏感，就会死。另一种猜想：A 板的电压受到了影响，导致信号接触不良（初步怀疑是 A 板的某些地方和车体框架导通，干扰了信号）	解决不了一点，能上的都上了，阀岛上轧带贴胶（上了两根，怕出问题），车周围上胶保护隔离，电路板，上稳压模块包胶	工程机器人 v2.0	嵌入式软件工程师： 盛李杰 机械工程师：张宸翰
17	装甲板导电	装甲板的螺丝孔肯定是和装甲板的地线接触，装甲板支架的地线不一定和装甲板支架的螺丝孔接触，所以说装甲板支架绝对会和地线导通	地线接通车壳	工程机器人 v2.0	嵌入式软件工程师： 盛李杰 机械工程师：张宸翰
18	灯条导电	灯条的支架和背部的铝方管连接，大寄	地线接通车壳	工程机器人 v2.0	嵌入式软件工程师： 盛李杰 机械工程师：张宸翰

1.7 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%）
李崇珊	机器人工程、大三、工程组组长&机械负责人	负责整个项目的项目管理；	30%

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%）
		负责工程机器人的机械臂及储矿机构的机械结构设计 负责整车气动系统维护	
盛李杰	机器人工程、大三、工程组电控负责人	整车电控代码编码及调试； 整车硬件布置和维护	30%
张宸翰	机器人工程、大二、工程组机械队员	负责底盘及救援机构设计； 负责跟进测试，维护机器人机械结构	20%
刘啸涵	机器人工程、大四、工程组电控队员	自定义控制器相关编码	10%
刘家荣 **（已离队）**	未选专业、大一、工程组算法队员	视觉识别兑换站位姿相关编码和测试	10%

1.8 参考文献

（实在没精力调格式了，谢罪）

《机器人建模和控制》，马克 W.斯庞

RM2022-南京理工大学-Alliance-机械结构开源-工程机器人，https://bbs.robomaster.com/forum.php?mod=viewthread&tid=22197

RM2022-广东工业大学 DynamicX 机器人队-工程机器人-机械结构开源，https://bbs.robomaster.com/forum.php?mod=viewthread&tid=22169

OpenCV 官方文档，https://docs.opencv.org/4.x/index.html

1.9 技术方案复盘

1.9.1 赛场性能表现情况分析

ARTINX 在复活赛-全国赛阶段一共打了 29 小局比赛，期间，工程机器人的表现较为平稳，为队伍提供了稳定的经济。一共出现过 2 小局取矿功能完全失能（提供 0 经济）的情况，是由于遥控器故障。期间我们的基本战术为开局直奔银矿，进行 3 连取然后回去兑换 3 个最高级矿石。

排除一些小局我们没有使用常规战术，一共统计到 21 次 3 连取数据，63 次兑矿数据。正常战术中我们平均每局花费 39.67s 取得 3 个银矿，平均花费 29.67s 兑换完第一个银矿，花费 33.35s 兑换完第二个银矿，花费 31.4s 兑换完第三个银矿（兑换第一个银矿会消耗在路上的时间，第 23 个需要等待兑换站运动）；综合来看，平均兑换一个最高级银矿需要 31.25s（算上兑换站运动时间）。在兑换的过程中掉落的矿石数约为 5 个，都是由于操作手误操作导致。

期间自定义控制器有极少局数是失效的，基本上是由于官方的串口线的问题，换自己的串口线后表现稳定。视觉辅助功能，由于我们的运行逻辑，在半数的局内是失效的，可能是操作手的位置不对，或者视觉解出的位姿超过了电控的安全限制。

比赛期间几乎全部进行最高级兑换，只有在热身赛等时候进行了个位数的次高级兑换。

机械结构和电气的外壳保护发挥了良好的作用，在比赛每一大局的间隔期间，没有发现结构失效或电气部件受损导致大修的情况。比赛的小局中间，也没有发生损坏导致下一小局不能上场的情况。

比赛期间一共发生了一次翻车。在对阵交龙的一小局中，工程在下 9° 坡时急转，发生了侧翻现象，索性没有造成结构损坏，下一小局正常上场。

在对阵大交 TOE 的一小局中，意外的发现工程抬高的图传机构可以强力的干扰哨兵识别击打前哨，利用工程的伸展尺寸来影响对面的视觉识别，可能是一个后续可以研究的功能。

1.9.2 赛场性能表现与规划对比分析

规划功能和预期表现	赛场性能表现	是否符合预期
底盘全向移动 上层机构时刻保持稳定，不翻车	移动和稳定性能良好 重心过高，加了速度限制后，有较小的翻车风险	基本符合

能稳定获取银矿，30s 内获取 3 个矿石	加上赶路和对准的时间，平均每局花费 39.67s 取得 3 个银矿	基本符合
能稳定兑换最高级矿石，30s 完成单次兑换	能稳定兑换最高级，实际兑换约为 20s 内，算上兑换站运动时间平均 31.25s 兑换完一个最高级矿石	符合
储矿机构稳定储存矿石，不卡矿不掉矿	没出过问题	符合
车壳坚固，保护完善	没出过问题	符合
拥有救援己方机器人能力，2s 内完成固连	比赛前效果不好，此功能砍掉	不符合
图传增加自由度，背面作正面开车	和救援功能一起砍掉	不符合

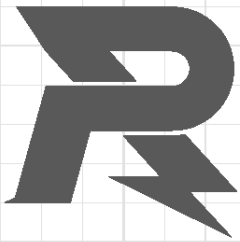
1.9.3 经验总结

整体来看，今年造出了一台能稳定拿 5 级银矿的工程车，在国赛期间贡献了稳定的金币数据，支撑起了我们的整体战术。尽管受规则改变之痛，功能上没有达到预期的效果，但整体上场的稳定性还是达到了预期。

这台车最难的地方是整体方案的成型。我们一开始提出一个前所未有的方案，奠定了他的路一定会前所未有的难走，确实是这样，前期反复的理论计算仿真，到实物的精度测试，到整车的策略和整合，这都是前所未有的挑战。我们也收获了前所未有的全新经验。

工程组的管理上，个人认为觉得好坏参半。开局时把最核心的几个机构分开拿给新人做，是组长的有意为之，只是确实结果证明对进度来说算是失败了而已，但我觉得还是一个必要的尝试。后来工程组的几位新人的成长速度也很快，后来工程组留队的人中，不管是去支援其他组的，还是留在工程组的，都发挥了主导性的或者关键性的作用。对于队伍管理来说，对如何给新人分配任务和如何评估项目进度也积累了宝贵的经验。

虽然规则把矿石数量砍了，导致赛季初的 5 矿梦想破灭，机械臂化身小丑，但现在看来，这个方案还是有独特的正确性的。减少取矿兑换的机构的空间，后面工程车的可能性会无限延展。但传不传的下去另说，未来还是要根据实际情况和具体规则再细细定夺明年的方向。



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