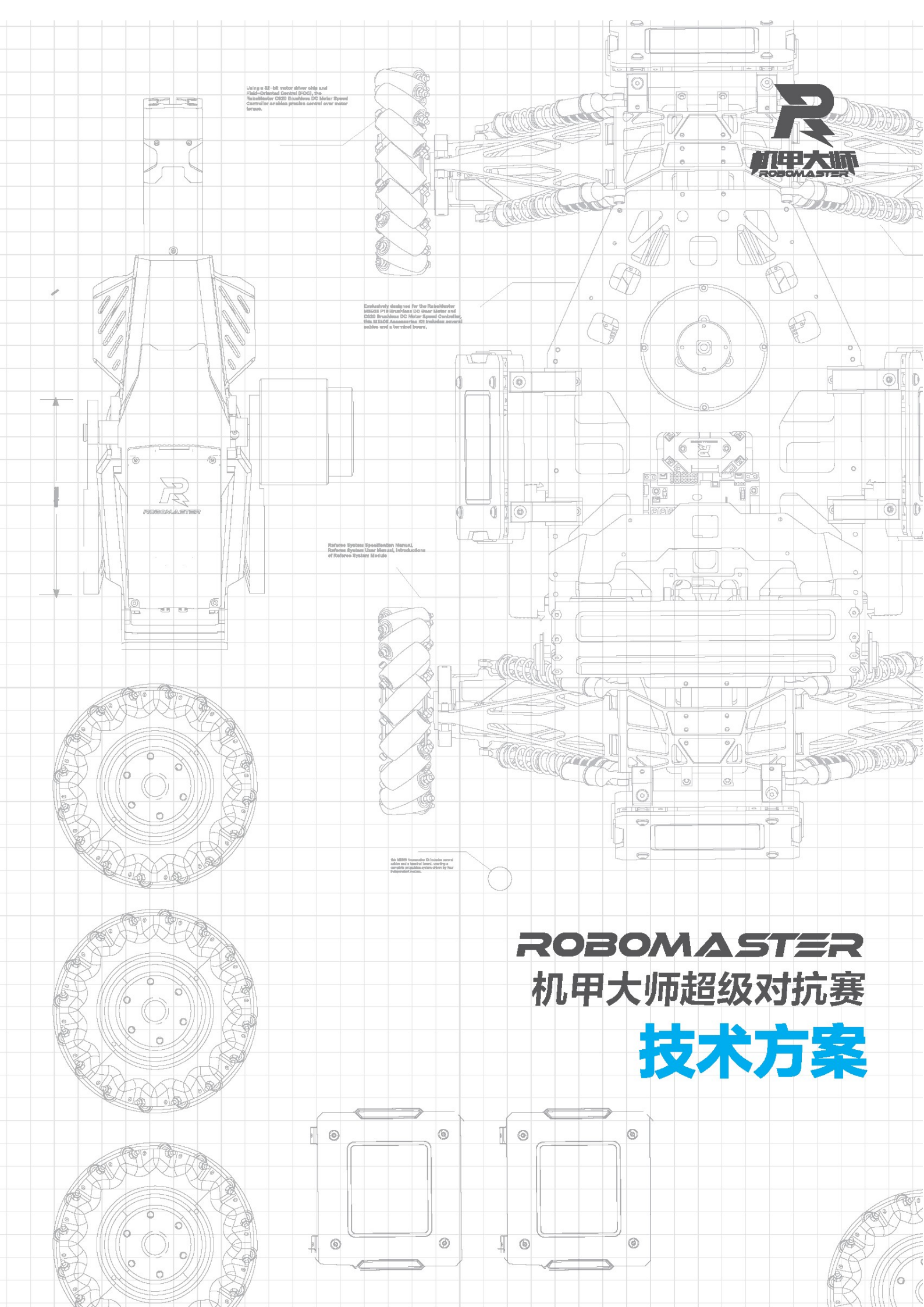


ROBOMASTER
机甲大师超级对抗赛
技术方案

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	刘乐，张宸翰，林沛君，肖星辰
硬件	盛李杰	吴彦辰
软件	盛李杰	
算法	刘家荣	

目录

1. 概述

1.1 背景&目标

本项目是一台可搬运特点道具的移动机器人，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，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。

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）

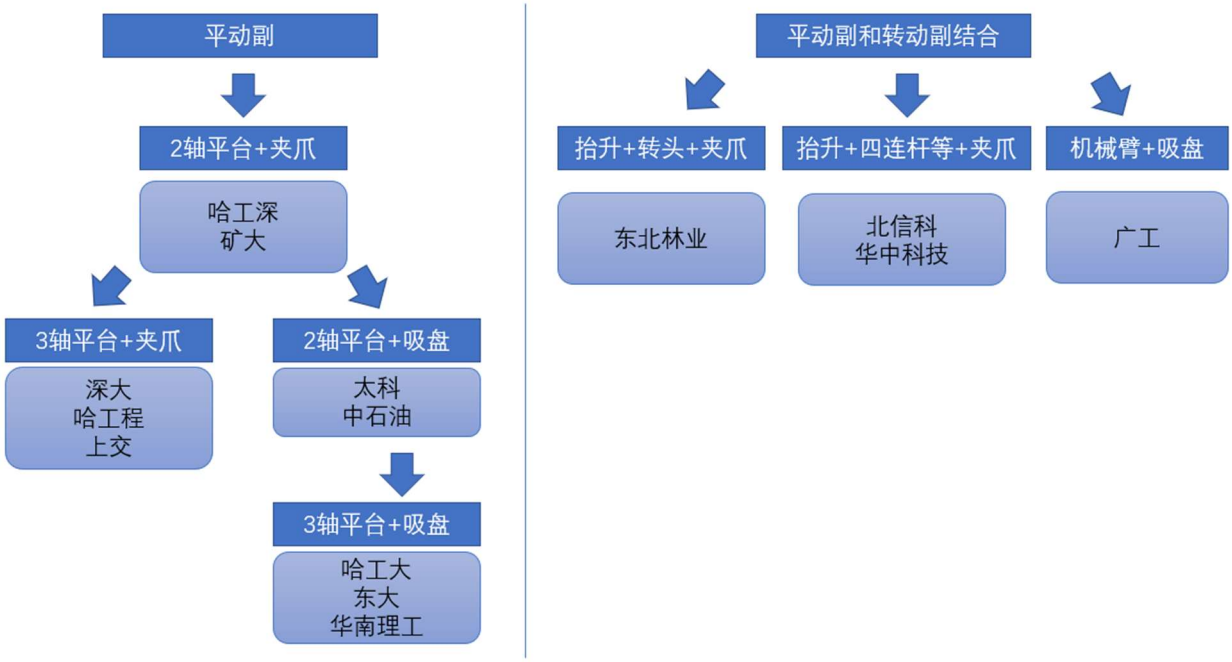
广东工业大学更是重量级，在 21 赛季做出了纯旋转关节（不算末端执行器）的机械臂。广东工业大学在 21 赛季虽然进入了全国赛，但工程机器人的表现说不上是十分的稳定（赛场表现看着感觉当年并没有上解算和路径规划？）。尽管如此，这样大胆的尝试着实是让人眼前一亮，在谁也想不到后续规则走向的 21 赛季，着实是让人产生油然的尊敬。

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。

1.3 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。

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。

1.3.2 整体功能定义

23 赛季工程机器人具体功能定义如下：

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 ● 5s 内完成一个矿石的获取 能够完成最高 4 级的兑换 ● 30s 内完成单次兑换 能视觉识别兑换站位姿，完成自动兑换 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 4 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

1.4 机器人核心参数

名称	*参数*
重量	*28.3kg*
收缩尺寸（长*宽*高）	*590*590*585*
伸展尺寸（长*宽*高）	*590*590*980*
执行器数量	*M3508 电机：8* *M2006 电机：2*

	GO-M8010-6 电机：2 *直线气缸：6* *真空发生器+吸盘：2*
最大移动速度	*2.5m/s*
最大矿石持有数	*5*
气瓶工作时储存气压	*≤20KPa*
气瓶容量	*0.8L*
工作气压	*0.3-0.5KPa*
工作真空度	*-41KPa*

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。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。

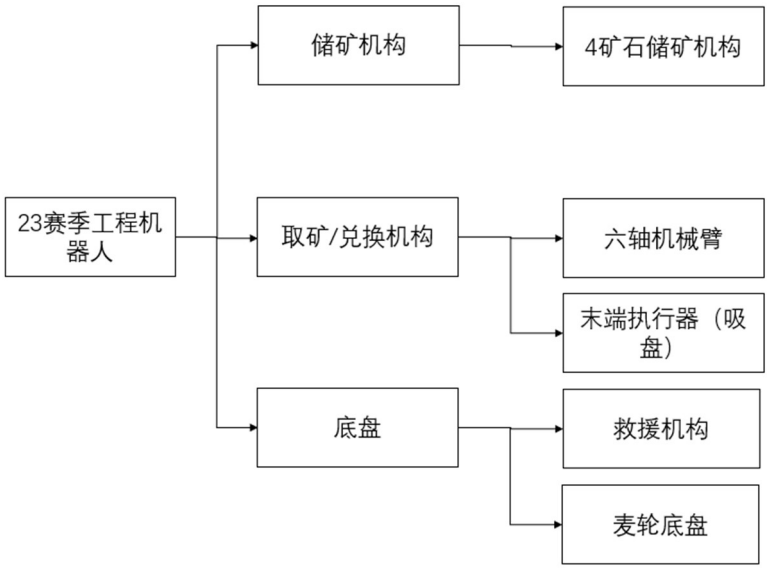
储矿机构部分，我们充分使用横向和纵向的空间，利用传送带来移动矿石的位置，使矿石储存的时候呈 2*2 的分布。为了满足基础尺寸的要求，储矿机构的初始形态是收缩的，比赛开始后，通过 4 个气缸进行两级的伸展，变为正常的形态。

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，且能够更充分的利用机器人上的正压气源。

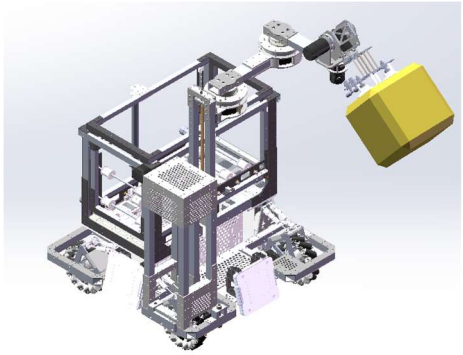
除了以上的核心机构之外，还有底盘和救援机构的部分。对于底盘设计，考虑到取矿时最好需要全向移动功能，考虑仍然使用麦轮方案。为了减少底盘对机械臂运动的影响，准备根据后续实际样机表现考虑是否增加底盘完全锁死的机构。救援方案采用传统拖拽救援思路，搭载于底盘上。使用钩爪+单向门结构结合。本体通过旋转的方式进行释放和收容，单向门的结构使得操作手的操作更加简单，避免了传统旋转钩爪操作时可能会把待救援机器人撞走的缺点。

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。视表现第二版样机可能加入完全锁死机构。

	救援机构	旋转钩爪+单向门机构。使用钩爪本体旋转的方式进行收容，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；末端并排两个吸盘，从侧面吸取小资源岛的矿石。
储矿机构	4 矿石储矿机构	利用竖直空间储存矿石（4 个竖直排成两行两列），用履带传输实现移动矿石，矿石从两个固定位置进出，有利于机械臂运动规划。



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/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，

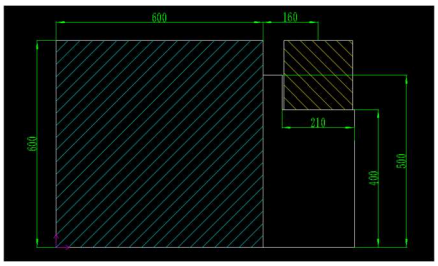
整理需求如下：

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

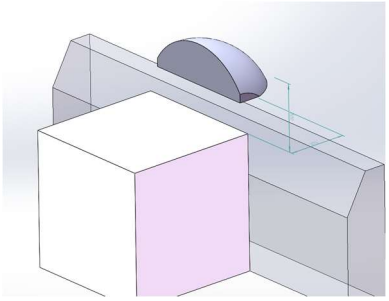
目标空间

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）：

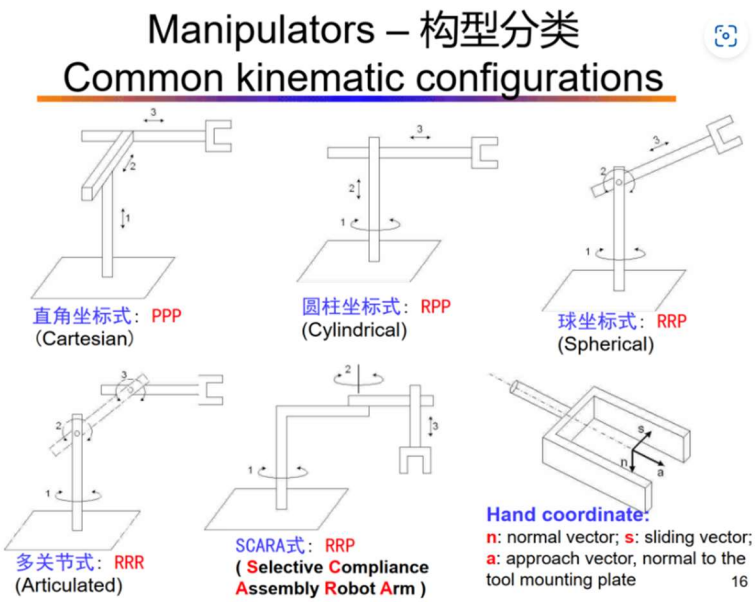


自身储矿机构的目标空间在车体内，且可以根据机械臂的构型调整位置，所以这里主要先考虑上面两个目标空间。

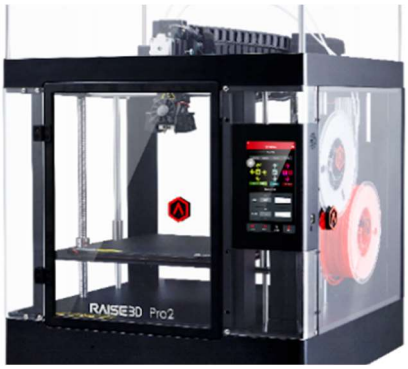
构型选择和比较

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：

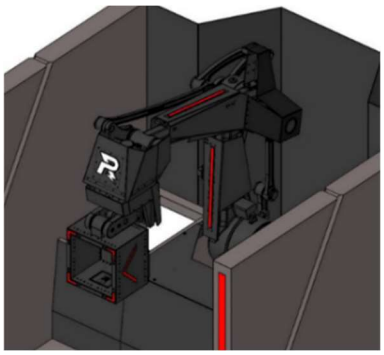


PPP：经典平动三轴平台，其优点是可采用成熟的滑轨抬升机构（开源一大堆，我们队伍也有较为成熟的技术积累），运动较为稳定，且运动解算极为简单；缺点是平动机构浪费车体的初始空间大，整机重量也高居不下，重心高更易翻车。工作空间为矩形，针对规则的变形空间利用率最高。



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。

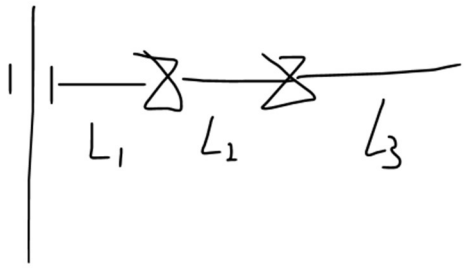


经典 6R 构型

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。

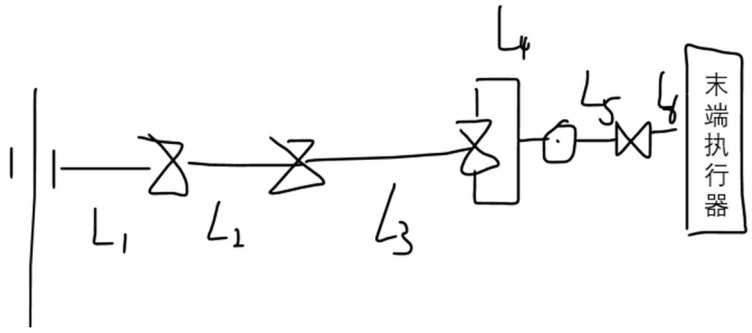
SCARA 型机械臂

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：

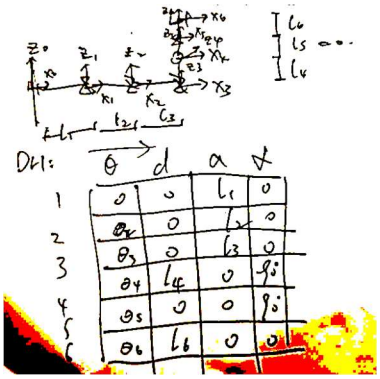


PRR（SCARA）+RRR（球形手腕）的六轴机械臂

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：

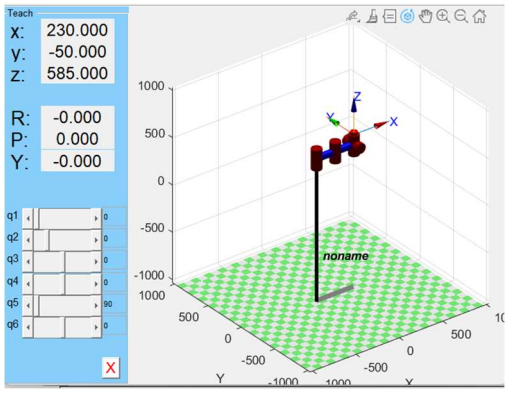


坐标轴方向与 DH 参数（注意球形手腕：l5=0）

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];
```

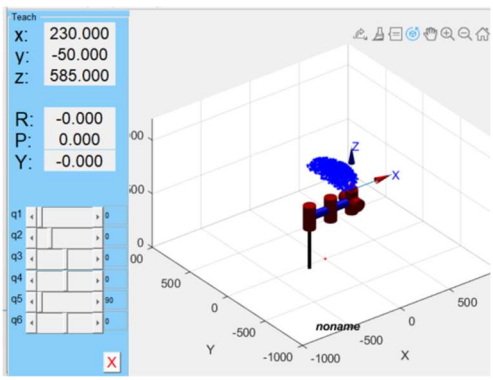
```
% DH 参数  [theta d a alpha]
L1=Link([0 0 L(1) 0]);
L1.jointtype='P';
L2=Link([0 0 L(2) 0]);
L3=Link([0 0 L(3) 0]);
L4=Link([0 L(4) 0 -pi/2]);
L5=Link([0 0 0 pi/2]);
L6=Link([0 L(6) 0 0]);

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

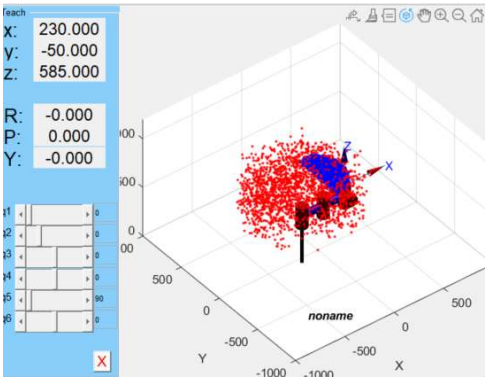
根据规则的数据，画出**兑换站**的目标空间：



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。

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。

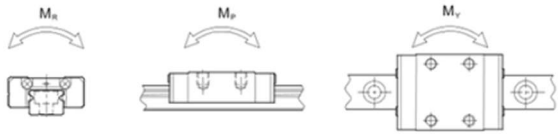
结构设计

第 1 关节（平动关节）

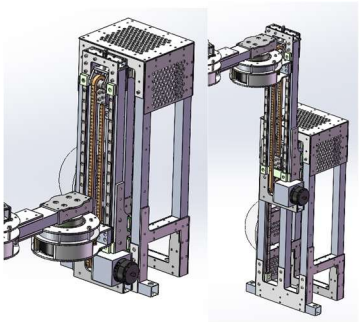
1 关节是一个竖直方向的平动关节。往后的 5 个关节质量之和约 5kg，且可能存在冲击工况，这里采用了负载较大的链传动来实现抬升功能；同时通过两套微型直线导轨来保证直线运动的稳定性，同时增强结构的刚度。

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，来保证运动的准确性。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）。

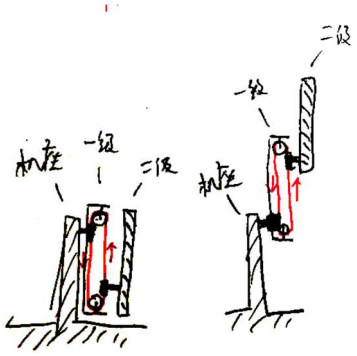


不同的静额定扭矩方向定义示意



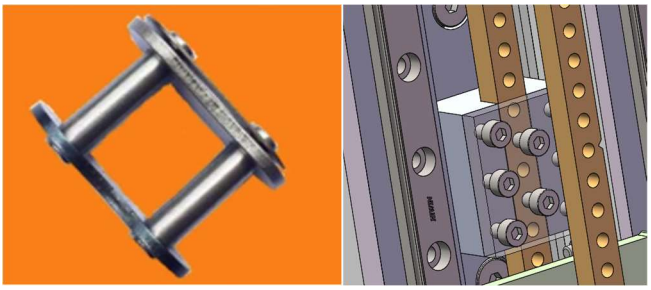
两段滑轨+滑块会同步进行抬升

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。

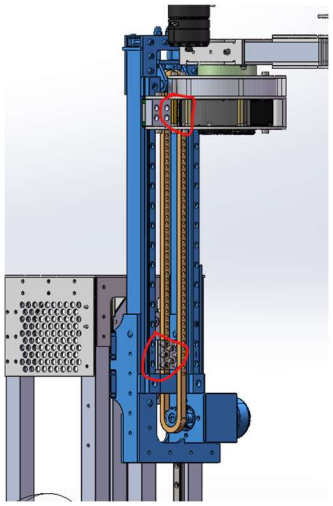


二级同步抬升示意图

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。

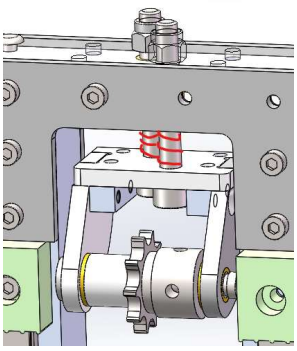


链条的固定



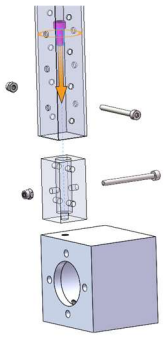
两处链条固定处

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 U 型的链轮架在样机阶段为板材拼接的结构，最终版会使用钣金件来保证强度。



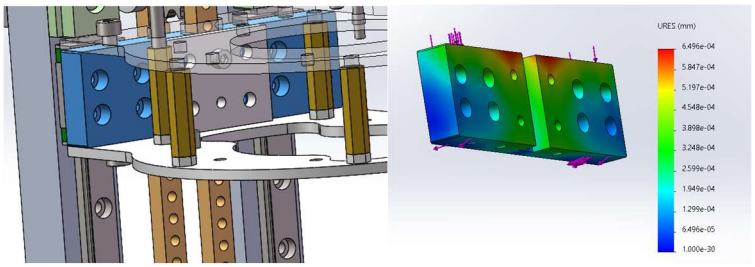
张紧结构

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。正面和上下有螺纹孔，在作为夹板卡住链条的同时，与滑块和第二关节的上下夹板连接。

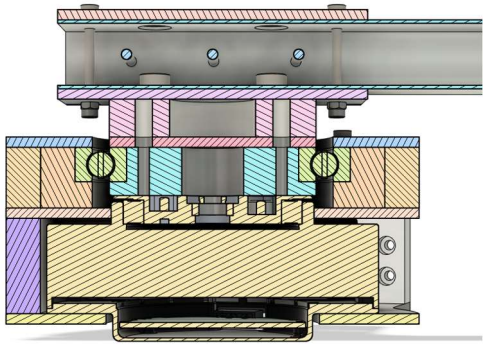


一二关节连接处

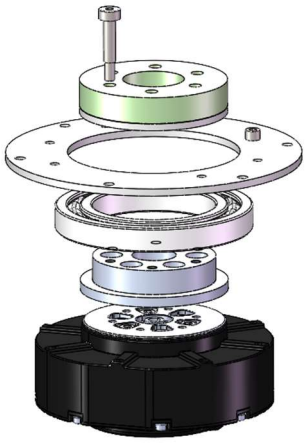
第 2/3 关节

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。

轴系截面图和爆炸图如下。



轴系截面图



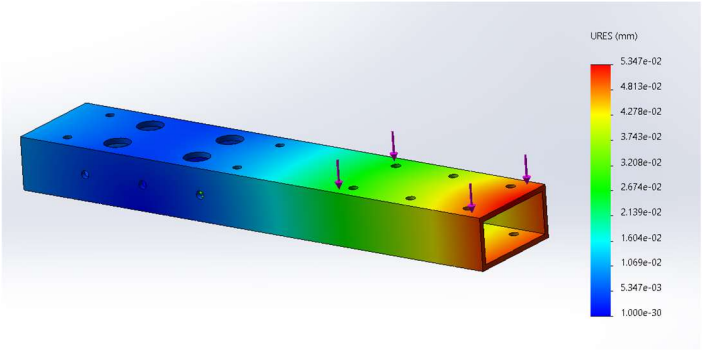
轴系爆炸图

电机采用宇树的 GO-M8010-6 电机，轴承皆采用 16010 深沟球轴承，经校核满足安全需求。关节二和三的连杆长度都为 215mm。

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，另外设计一工件实现延长输出法兰的作用，考虑到轴承的尺寸和安装的空间，另外设计一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 6 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。

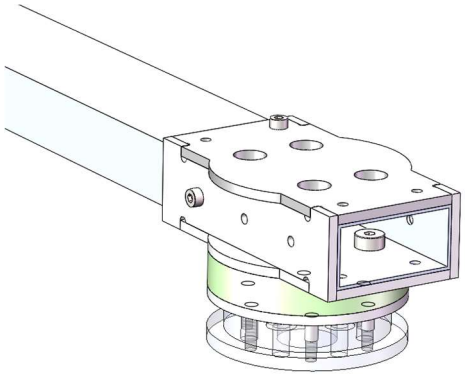
电机与延长输出轴的工件配合后

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

连接处在方管的四面加上了玻纤板，并用螺丝拉紧，这样把螺丝的应力平均到铝方管的表面，增强连接处的强度。

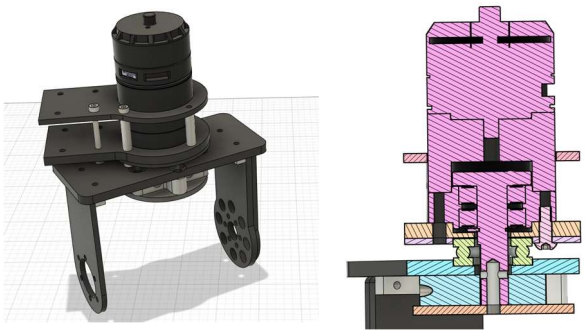


关节连杆连接处

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

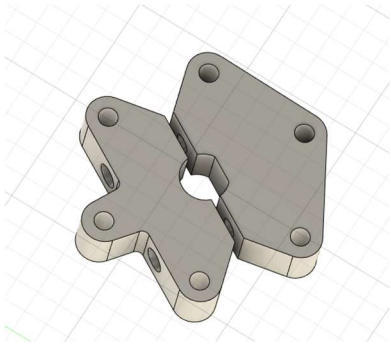
第 4 关节

第四关节采用大疆的 M3508 电机，电机和连杆间安装了一个推力轴承。这里型号为 51101 推力轴承。



关节 4

电机的输出轴为Φ10 的 D 型轴，这里采用（某个已标准化上架某宝的）分体的法兰工件来固定 D 型面，成本更加低且更方便拆装和更换。



法兰联轴工件

第 5 和第 6 关节

第五和第六关节皆使用大疆的 M3508 电机，第五关节使用两个较薄的非国标深沟球轴承。负载较小，轴系设计时以节约空间为主要目的，因此采用非标的薄轴承。

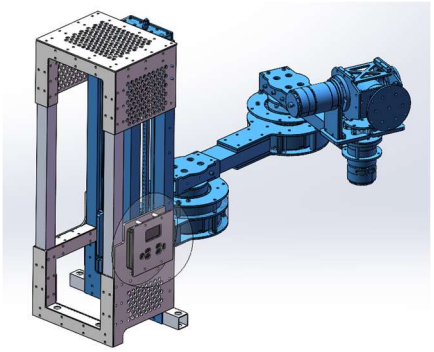
第六关节直接与末端执行器连接，负载较小，不使用轴承。

第五和第六关节的设计借鉴了 2022 广东工业大学的末端轴系，在此表示感谢。

关节 56 截面&爆炸图

机座

机座部分由铝方管和玻纤板铆接而成，然后铆接到关节 1 的机座上，下面有四个 M10 的螺纹间隙孔，通过 4 根 M10 的螺丝连接到底盘，具体定位的零件详见底盘的部分。



机座架

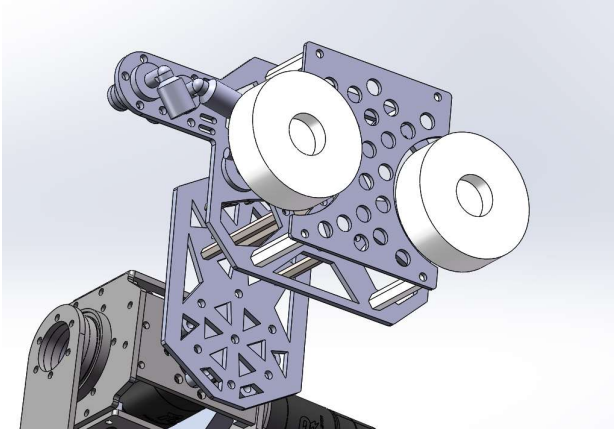
1.5.1.3 末端执行器

整体设计

本车末端吸盘需要完成取矿、兑矿的任务，且依据本赛季的经济规则，评估认为取银矿效益更大，所以吸盘末端的设计着重在吸取矿石时的稳定性和鲁棒性，而不考虑往年赛季使用吸盘进行空接。

在设计初，规则还没有改为底盘可移动，因此使用了六轴机械臂，这也要求末端尽可能减重。

（该版本设计从侧面吸取矿石）



核心结构设计说明

末端有三个设计板块，分别是吸取、固定和走气部分。

吸取

因为本赛季采用了六轴机械臂，末端执行器的重量不宜过重，且矿石兑换框左右仅有 20mm 余量，如果使用夹爪，兑换框两侧余量会进一步被压缩，对位置识别有更高要求。考虑到以上两点，我们选择了真空吸附系统进行矿石吸取。

真空系统中采用了减压阀、电磁阀、真空发生器、真空过滤器和吸盘元件：

气动元件	型号	店铺
减压阀	ARM5SA-08A	金茂自动化
真空电磁阀	[1 位底座] G 型 24V	欧思托气动
真空发生器	ZU05L	杰安德气动元件
真空过滤器	VFJ66	杰安德气动元件
吸盘	双层 60mm 海绵吸盘	鑫悦自动化配件

气动元件型号

真空系统选择

按真空产生的方式分类，真空系统可以分为真空发生器系统和真空泵系统。两者比较如下图所示，真空泵有很多种类，但是体积够小，能在赛场上使用的真空泵一般是机械泵，以下比较均以机械泵举例。机械泵是利用机械方法，使工作室容积周期性地扩大和压缩产生真空；真空发生器则利用高速射流卷吸走负压腔内的气体，使腔内形成很高的真空度。

	真空泵	真空发生器
最大真空度	可以同时到达最大值	不能同时到达最大值
最大真空流量		
体积	大	小
重量	重	轻
驱动	电力	压缩气体
寿命	有可动件，寿命较长	无可动件，寿命长
维护	需要	不需要
真空产生与解除	慢	快
真空压力脉动	有脉动，需设真空罐	无脉动，不需真空罐
应用场合	适合连续、大流量工作，不宜频繁启停	需要提供压缩空气，宜从事流量不大的间歇工作

两种真空发生装置对比

因为需要通过机械运动产生真空，所以使用时真空泵不可避免会有震动，不过优点是真空泵使用电力运行，不用担心中场失效。真空发生器最主要的优点是体积小，重量轻，使用时无震动，可以放置到离末端较近的地方，快速产生真空，但是真空发生器需要使用压缩气体驱动，需要在工程车上安置较大的气瓶保证供气。

考虑到真空泵的震动，我们决定优先使用真空发生器，如果测试发现耗气量确实太大再进行更换。因为不需要空接，所以对吸盘的响应时间要求不高，这里主要验证不同种类的真空发生器对耗气量的影响。

真空发生器分箱式和管式两种，箱式的真空流量和最大真空度都更大，但是重量和体积也比较大，如果要使用需要固定到车上，而相同性能的管式重量轻，仅比气管直径大一点，可以在任意没有弯转的气路

部分使用，因此优先选用了管式的真空发生器。

此处我们对管式和箱式进行了实际比对，测试数据详见 1.6.1 *表真空发生器在 0.3L 20MPa 供气时使用时间对比*，可看到尽管箱式在更小的供气气压下就能吸取矿石，但是工作时间还是完全无法和管式比。

数值计算

气瓶内气体的减少速度和输出气压有关，要确定真空发生器的有效工作时间，还需要确定气瓶输出的气压。依据真空发生器的排气特性，气瓶输出和真空发生器的真空压力一一对应，因此确定真空压力即可确定供给压力。

理论公式：

$$W=\frac{\pi D^2\times n\times p}{4t} \qquad (1)$$

式中，D 为吸盘直径(mm)；W 为物料重量(N)；

t 为安全系数，水平 $t\geqslant 4$；p 为吸盘内的真空度(MPa)，$p=(63\%\sim 95\%)p_v$；p_v 为真空发生器最大真空度；

数值估计：

根据真空发生器计算公式，要确定供气压力需要获得以下参数：1.矿石质量 2.矿石与吸盘间摩擦系数 3.安全系数 4.吸盘直径

1. 矿石质量：

   根据 22 年规则手册，矿石质量在 0.6~0.8kg 间，为保险，我们选取 0.8kg 计算。

2. 摩擦系数：

   矿石材质为 EVA 发泡塑料，海绵材质为 EPDM 橡胶，查询材料配对摩擦系数表时未查询到对应数值，于是采用相似材料的进行估算，第一列数值为静摩擦系数，第二列为动摩擦系数，拟定摩擦系数为 0.5。

橡胶	橡胶（平行纹理）	0.62	0.48
橡胶	橡胶（交叉纹理）	0.54	0.32

3. 安全系数：

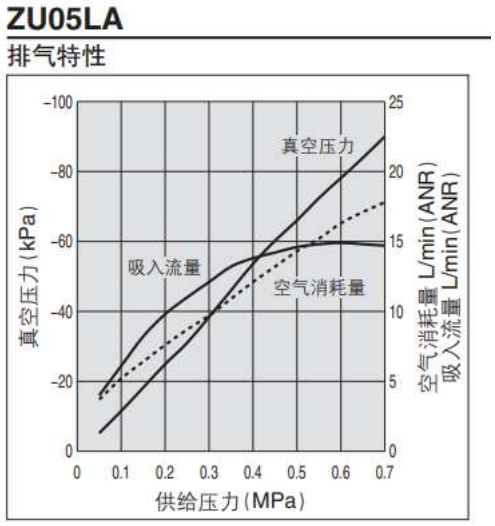
   竖直时，安全系数 t 应该大于等于 8，通常情况下取 t=8

4. 吸盘直径：

   根据侧吸要求选定吸盘，直径尽可能大，减少耗气，但是 60mm，数量为 2，即 n=2

- 结果：

  $p = 4tW/\mu n\pi D^2$ = 0.02218 MPa = 22.18 kPa，根据理论计算，真空度只要达到 22kPa 即可使用



左图是 ZU05LA 的排气特性，从图中找到 22kPa 真空压力时的供给压力在 0.2MPa 左右。

依据该计算结果，我们进行测试验证，测试时我们使用了 0.3L 气瓶，充入 20MPa 压缩气体，对 SMC 的标准型大流量 ZU 进行测试，测试的真空发生器型号如下所示：

型号	供气压力	最高真空压力	最大吸入流量	空气消耗量
ZU05L	4.5bar	-48kPa	12L/min	14L/min

真空发生器参数

先验证 0.2MPa 的供气压力是否能在一定冲击下吸附矿石，具体测试结果可以在 1.6.1 表 *供气压力测试* 中查看，最终结论是在 0.25 MPa 以上的供气气压，可以稳定吸取矿石。

因为供气压力的大小也会影响气瓶气体的使用时间，因此我们选取 0.25MPa 输出气压。

这里测试使用的是 0.5L 气瓶，压缩气体气压为 20MPa，输出气压为 0.25MPa，末端执行器是两个 60mm 直径的双层海绵吸盘，最终测试结果如下，时间有 5 分钟，减去场上工程车移动和吸取前位置调整的时间，还算有余。

气源表压 MPa	真空发生器	时长	真空度 Kpa
0.25	管式 05L	05:10.39	32

工作时间测试

根据以上所有分析、计算和测试结果，我们选用了 ZU05L 管式真空发生器产生真空，气瓶以 0.25MPa 供气。

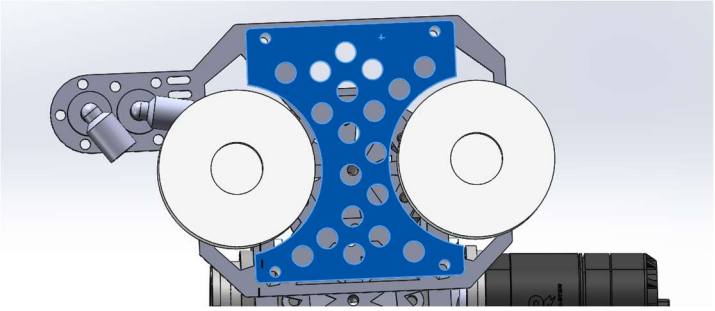
吸盘选择

考虑到赛场上有诸多不确定性，可能产生碰撞、被弹丸击中、光线变化导致视觉识别误差等意外情况，所以吸取部分采用了更保守的双吸盘结构。

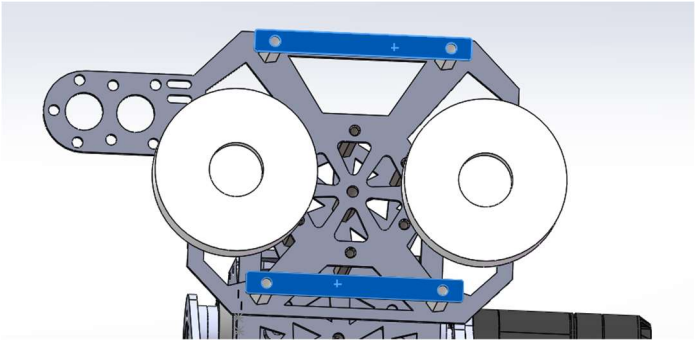
首先是吸盘的选型，因为要求末端轻量化，我们先排除了重型吸盘，之后经过比较，使用了双层海绵吸盘，相比平型吸盘，双层海绵吸盘能包容一定程度的不平整表面，如果在取矿时有偏差，吸到了矿石边角上，也能成功取到矿石。但是海绵有薄海绵和厚海绵两种，厚海绵对凹凸表面的适应性更好，但薄海绵质地密，所以我们猜想会不会薄海绵有更好的气密性，此处我们使用了两种厚度和大小的海绵吸盘进行测试，验证吸盘对真空度的影响，详情见 1.6.1 表 *相同供气气压下吸盘对真空度的影响*，可以看到两种吸盘虽然类型和大小都不一样，但是对真空度影响并不大，最大差值也只有 3kPa，而且也不是其中一个真空度一直更大，由此粗略推断，吸盘类型和大小对真空度都没有影响，所以最后选了鲁棒性更好的厚海绵。

再就是供气方面，两吸盘各使用了一个真空发生器，分别供气，一是保证吸力足够大，即使剧烈晃动也不会掉矿，二是即使一个气路出现问题，末端也能工作，只是没有两个吸盘那么稳固。

同时为了防止矿石产生以两吸盘连线为轴线的旋转偏移，设置了防偏板用来支撑矿石。不过只用来支撑的话仅在两边安装条状板即可，这里使用一块整板主要是为了保护气路，防止被弹丸正面击中。



现采用的连板设计



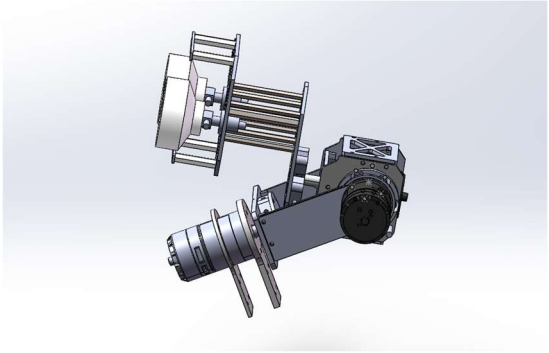
开始的条板设计

固定

吸盘固定板和电机联轴器之间用了一块偏置板，中间使用螺柱连接吸盘的固定板。

偏置板有两个用处，一个是使机械臂第六轴能够满足 0-60 度转动，该转动角度是由 matlab 仿真计算得出的能达到所有兑换姿态的最小角度，如果将吸盘固定板直接和联轴器固定，将会产生机械干涉，所以使用用偏置板转接；另一个作用是从侧面吸取矿石时，银矿石仅有一半露出资源岛，所以吸取矿石时机械臂第六轴轴心所对位置不是吸盘中心，不利于机械臂位置解算，偏置可以补偿该偏差。

螺柱的目的是为了配合资源岛前方的宽度，使吸盘能够到达矿石而联轴器不和资源岛干涉。



机械臂第六轴 60°时姿态三轴的位型

走气

气路不像电路，对走气路径要求更高，不能产生弯折或者拉伸，不然很可能出现气流不畅或者接头松动的情况。因为机械臂前三轴的结构是 PRR，且 RR 旋转轴平行，走线走气均不会产生弯折扭转，所以这里仅写了后三轴的走气设计思路。

后三轴是结算简单、较常使用的三轴垂直相交的结构，但是显然气路不能贴在三轴上固定，很容易产生扭转，所以这里将气路设为外走线，即直接从第六轴伸出到第三轴，从而避免气管随电机的转动。

吸盘金具使用侧走气，走到吸盘固定板上一个突出位置，连接到隔板直角快插接头上，既固定了气管，也在不弯折的情况下转变了气路方向。之后在气路外加波纹管进行保护，连接到第三轴上。

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 4 个矿，加上吸盘自己带一个可以总共完成 5 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；

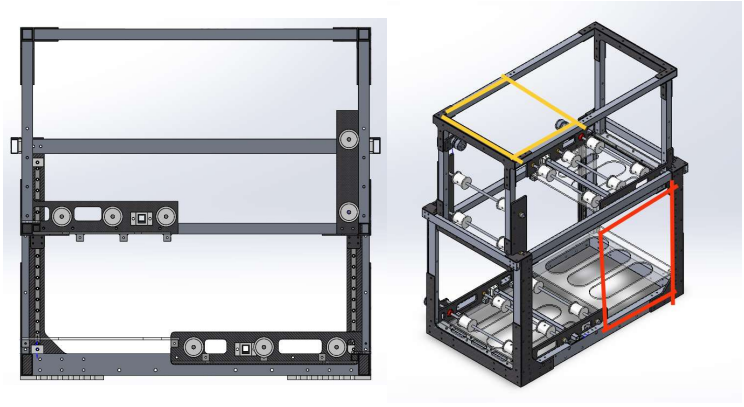
3. 整体结构稳定安全，不会出现因撞击等丢失矿石或卡住的情况。

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

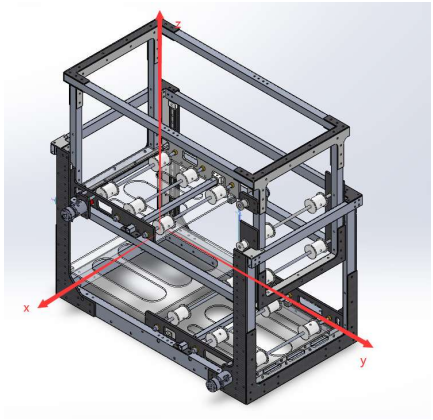
基本原理

为方便放入及取出矿石。我们尽可能做到：多次的放入和取出矿石位置不变。这需要我们做到用传动装置将矿石移动。参考方形物体移动的实例，传动装置为同步带传动。



储矿机构示意图（黄色为矿石入口，红色为矿石出口）

基本尺寸

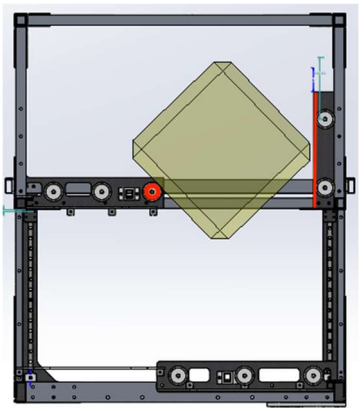


储矿机构示意

因矿石为 200*200*200mm，及储矿机构尽可能占用较小空间，将矿石活动的 x 间距定为 220mm。

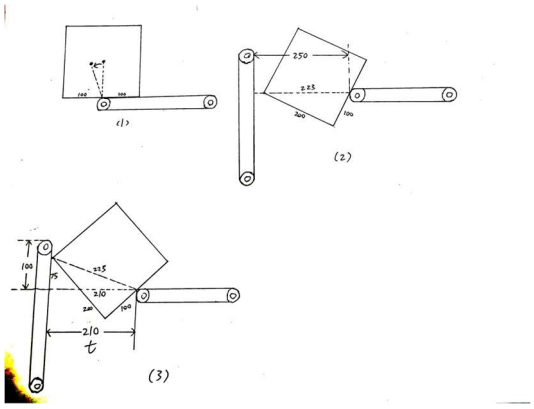
储矿机构极限宽示意

储矿机构占用空间较大，为方便矿石移动及由上层下落。我们由底盘设计及制作规范手册中的装甲板遮挡范围，确定外框架外侧(左图两红线)尺寸为 482mm



矿石翻转示意

圆心和线间大于：同步轮半径+√((82)²+(200)²)，这样矿石可旋转 90° 掉落。



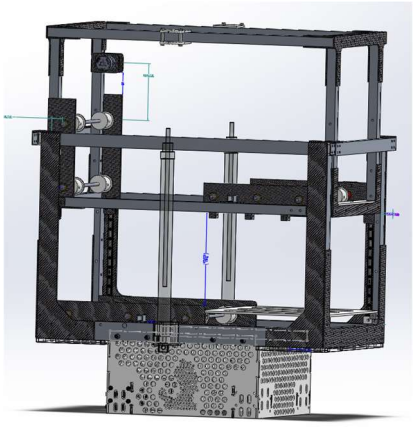
翻转相关计算与示意图

抬升机构

工程车有最大初始尺寸限制，因此我们增加了两级抬升机构来使储矿机构的初始尺寸满足规则要求。

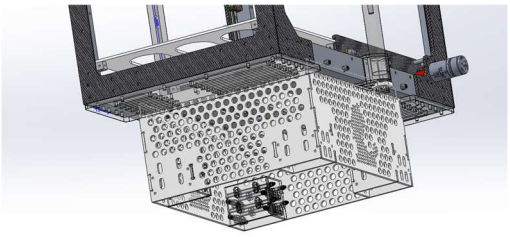
- 一级抬升：把储矿机构整体抬升，使出矿的口到达机械臂的工作空间内。
- 二级抬升：储矿机构内部展开，使上下层间距能容下矿石。

一级抬升用于将储矿机构整体抬升。只需要收起与完全展开两种状态，无需在展开过程中停止。因而选择了气缸作为动力源。

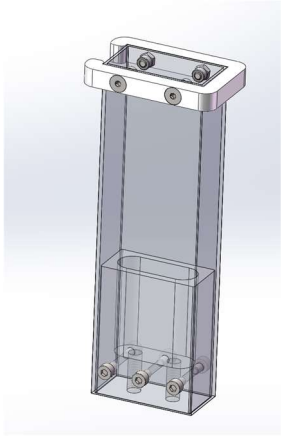


储矿机构抬升结构

储矿结构抬升部分由四根与底盘相连的铝方管作为抬升的行进轨道，同时在铝方管上也设计了物理限位，保证了抬升形成的可控可靠。

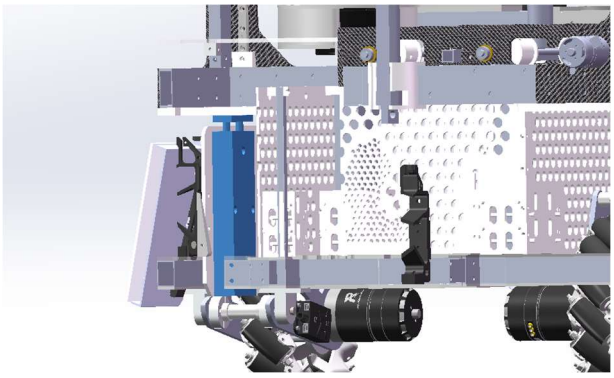


（仰视图）



作为支柱的铝方管及其固定结构

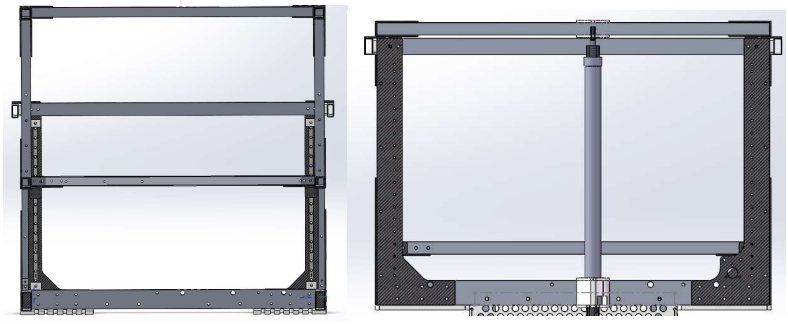
因为一级抬升对结构的水平程度影响最大，在一级抬升选择了使用并联气缸，以此保证抬升时的平行度，与抬升后储矿结构的水平，便于将矿石放入、取出储矿机构。



一级抬升处的并联气缸

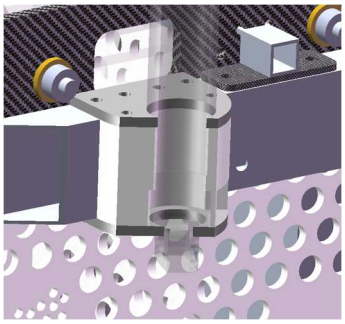
二级抬升在储存矿石时伸展储矿机构，完成同时容纳 4 个矿石的目的。抬升机构仅需保持两个状态：完全压缩的入场状态及完全张开的储矿状态。我们选择用气缸+滑轨的方式进行抬升。

由于储矿长边限制，气缸安装在短边上。为避开上图矿石取出点，我们对气缸略微偏移。



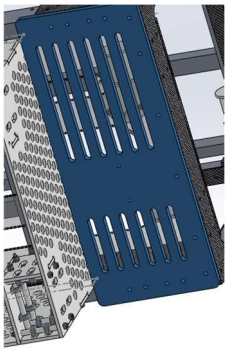
储矿机构内展开（左）和收缩（右）示意图

二级抬升因为在储矿机构内部使用了滑轨，保证了抬升时的平行度，因此选了长行程气缸。但因气缸上端不便于固定，为了保证抬升时气缸自身不会偏转、位移，使用 3D 打印件与板材与气缸下端紧密贴合，经测试，可保证抬升时气缸的稳定。



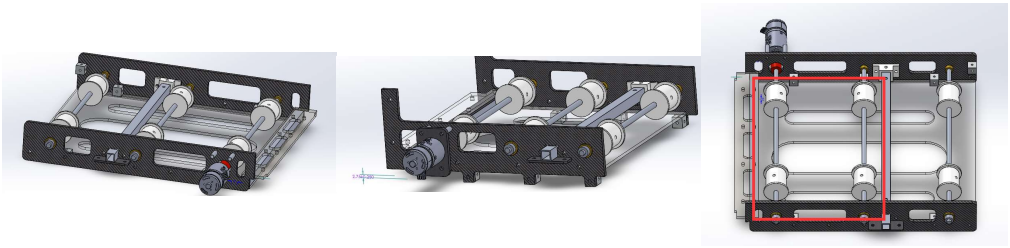
气缸底端固定处剖视图

抬升时并联气缸与储矿机构接触处需保证结构刚性，不能发生形变。因此选择使用 6mm 厚玻纤板，同时进行减重，保证储矿结构底部的刚性与重量。



储矿机构底部

传送带轮组设计



三个单独的轮组模块

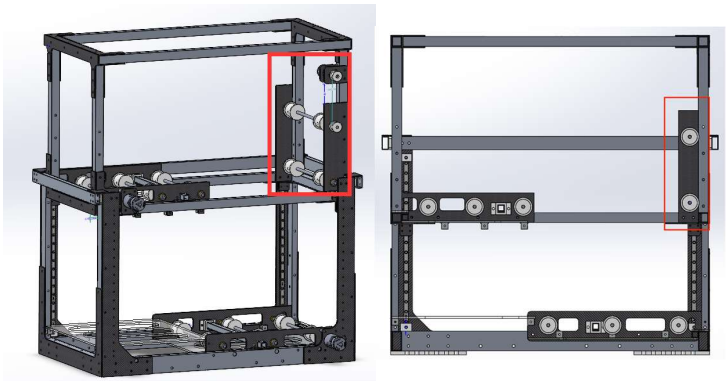
在设计和初次实践中，我们发现将固定轴的玻纤直接与框架铝方管连接平行度不高，加之采购铝棒不直，转动不理想，故采用将轮组模块化。在确定平行度的情况下，单独形成小框架，再与储矿机构的框架

装配。

考虑到矿石极易与侧壁发生碰撞，有垂直运动方向摩擦力，故采用有挡边同步带轮，防止脱轨。又需同步带与矿石接触，故采用双面同步带。

考虑到同步带轮直径及轮组框架空间，最终轮组高度为 45mm。

侧面轮组



侧面轮组位置示意

考虑到空间较为狭小且尽可能保证矿石在运动中姿态确定，同时矿石下落时不卡住，我们添加了侧面轮组。

若电机直接用联轴器连接，占用抬升空间，故电机安装在侧面轮组上方，不影响矿石运动和抬升框架运动。并使用另一组小同步带将电机动力传至侧面轮组。

1.5.1.5 底盘

需求分析

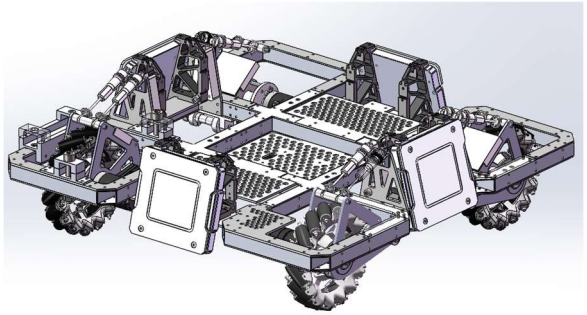
底盘是机器人上层机构的机座，同时机器人提供移动功能。在 23 赛季的规则中，起伏路段大幅减少，对于我们只取小矿石的取矿-兑换的路线来说，地面全是平整的地面，因此对底盘的减震性能要求并没有那么高。同时，在对矿时最好能有全向移动的功能。除此之外，上层机构（机械臂）运动时，会对底盘产生反作用力，可能造成底盘打滑位移，因此底盘还需要具有对上层机构运动的阻尼减震性能，最好还能相对地面完全锁定。

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）

考虑到研发时间成本和人力等因素，我们暂时不实现功能 3。这一版采用普通的麦克纳姆轮移动底盘方案。

结构设计

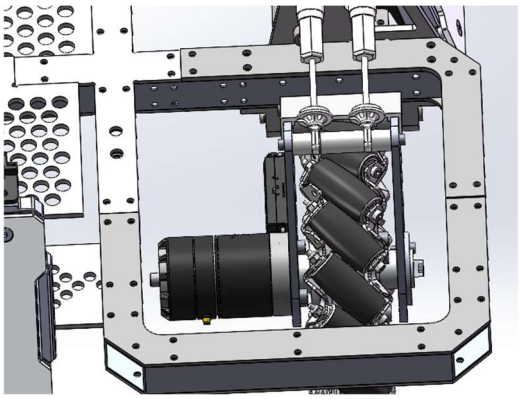


底盘图纸

底盘参数：

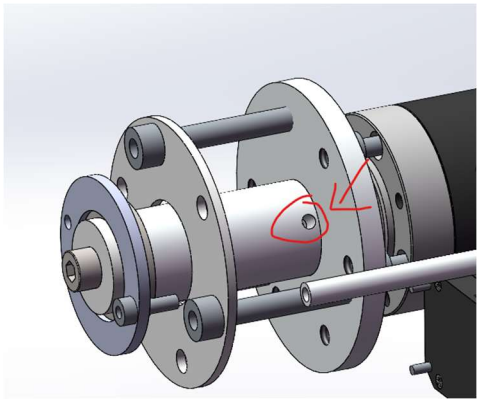
名称	*参数*
尺寸（长*宽*高） *注：高度指框架上平面到地面的距离，不考虑超过框架的装甲板等*	*590*590*124（mm^3）*
下框架离地高度	*100mm*
重量	*12.5kg*
接近角	*53.7°*
通过角	*53.6°*
轮距（横*竖）	*438*380（mm*mm）*

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



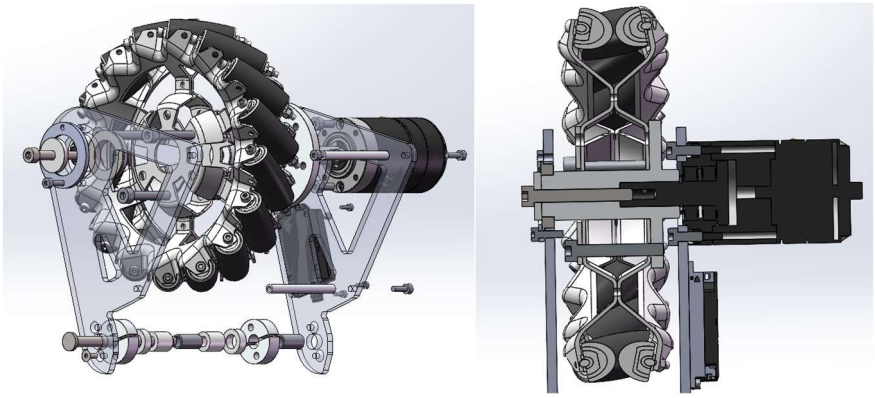
板材对齐保证装配精度

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。



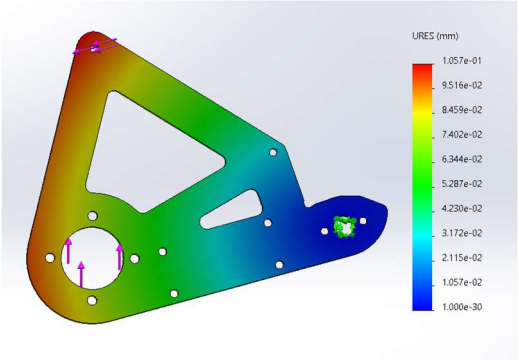
销轴位置

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

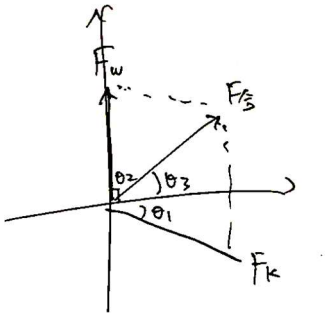
轮组两侧的夹板为两块厚度为 4mm 的玻纤板，经仿真最大位移为 0.1mm，符合要求。



轮组侧板仿真结果

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).

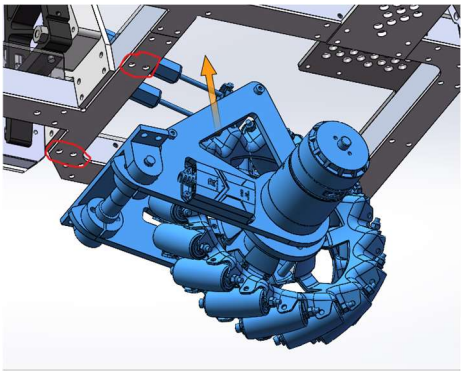
弹簧常数公式（单位：kgf/mm）：k=(G*d^4)/(8*Dm^3*Nc)

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

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

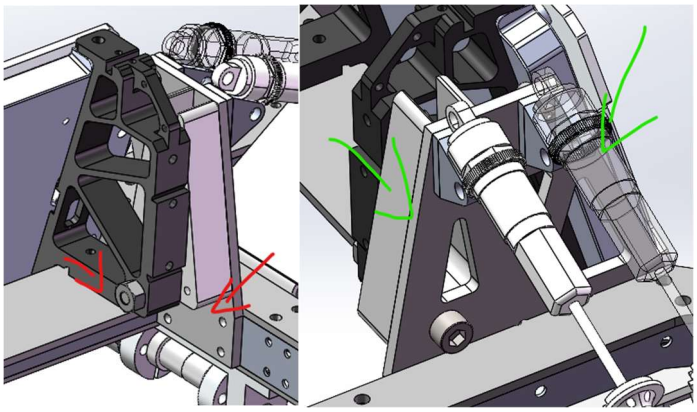
根据弹簧线径公式，选用不锈钢丝弹簧，代入相应尺寸数据，求得避震器弹簧线径 d=2.1mm。所以**选用线径为 2mm 的不锈钢弹簧**。

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。



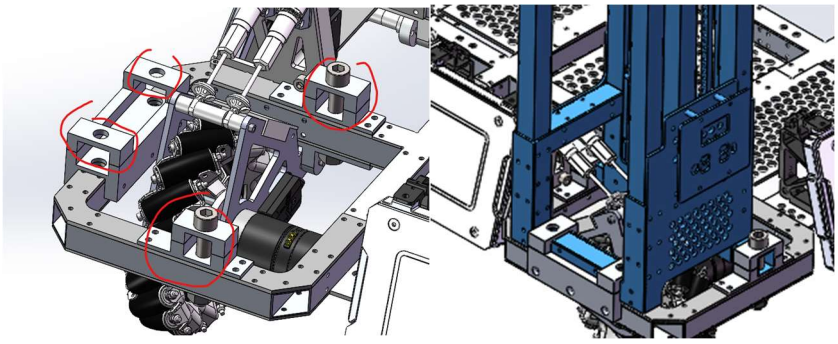
轮组和底盘的连接

避震器的另一端固定到一块板材上，这块板的背面铆接在底盘框架上，并且通过一根 M6 的螺丝与装甲板支架连接。机器人在地面运行时避震器只会受压而不会受拉，因此这块板的上面没有进行紧固连接，而是直接进行了两处支撑，如下面右图，左边为 3d 打印件支撑，右边为装甲板直接支撑。这样支撑之后，这块板的抵抗避震器收缩的抗弯能力大大提升，又利用了装甲板的刚性，同时使装甲板和底盘框架稳定的连接，个人认为是比较简单和紧凑的设计。



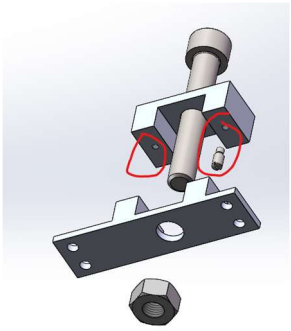
避震器座的紧固连接（左图）和支撑（右图）示意

底盘与机械臂的连接采用了 4 个 M10 的螺丝进行连接，稳固的同时能够较快的拆卸。



底盘与机械臂连接处示意

在每个 M10 螺丝的固定处，由两个加工件来与机械臂机座的铝方管的四个面配合，上方的机加件有两个螺纹孔，下方的机加件有两个定位孔，上下两个机加件的由两个轴位螺丝来定位。这样定位较精准的同时又方便安装和拆卸



装配顺序和轴位螺丝示意

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

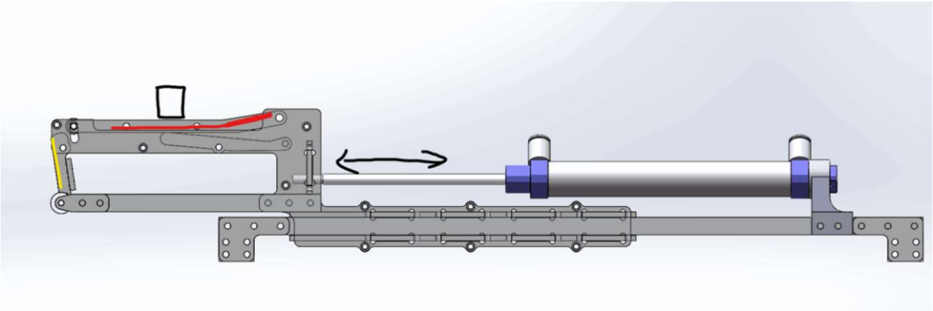
1. 抓/钩取容易、分离方法简单。
2. 操作手操作简单，不需要精细的位置调整。（允许操作误差）
3. 寿命长，拆卸、更换简单。
4. 占据小空间，不影响工程车与被救援机器人其他机构的设计、摆放。
5. 动力源需求少。

结构设计

通过预调研，发现历届队伍开源中，救援机构主要分为四种策略：抓、钩、电磁铁、单向门。分析后，为了满足抓取分离容易，降低对操作手的操作精度要求等需求，选择了本队在往年使用的单向门方案，并根据往年比赛的经验加以改进。

单向门结构，如其名只能使物体单向进入，不允许物体离开，满足了抓取的要求。再通过释放结构，便可满足与抓取机器人的分离。同时，单向门结构在救援时可以提前伸出救援机构，通过与被救援车辆的碰撞完成抓取，无需在被救援车辆前停留，保证了救援快速与救援时工程机器人的血量安全。

本队往年救援机构分析



往年救援机构结构

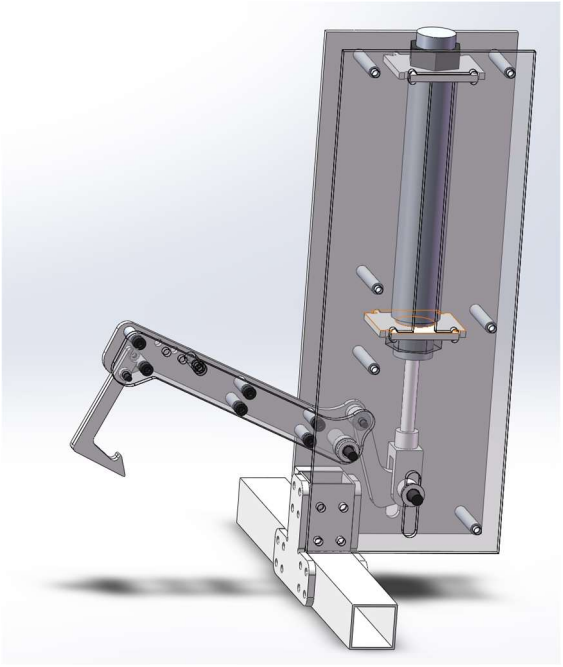
通过气缸为动力源，推动机构在水平方向运动。在气缸伸出式时，连杆与上方铝方管相互挤压摩擦，使黄色部分形成单向门。在气缸收回时，连杆归位，使黄色部分可以自由旋转，完成释放。

通过此种方式，有简单、高效的优势。但通过与方管摩擦实现功能代表着其寿并不能保证。而且此种方式占据底盘过多空间。

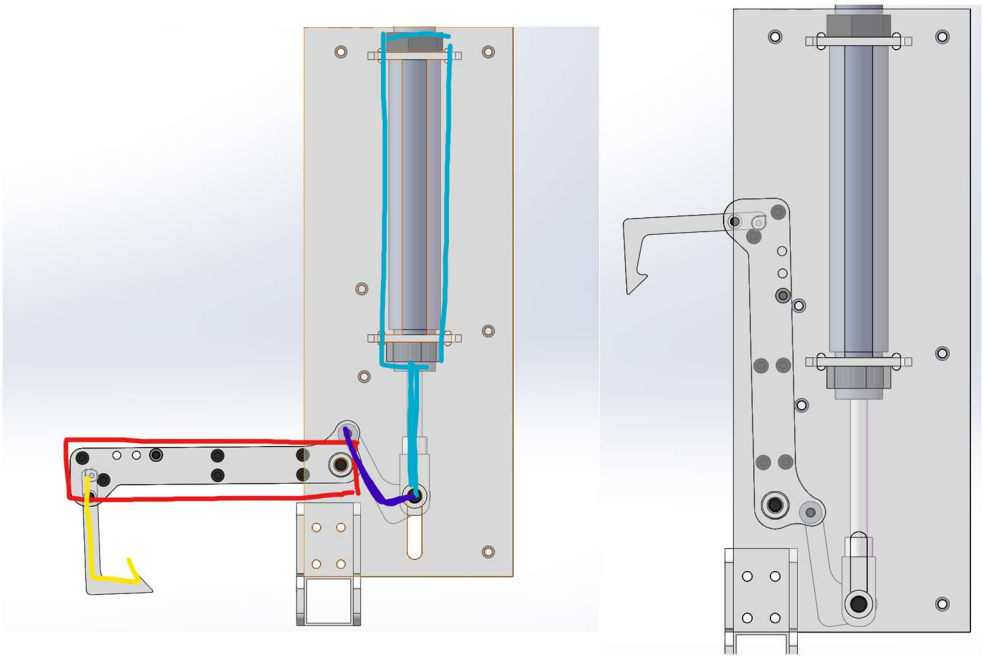
救援机构设计介绍

为了设计一种更可靠，寿命更长且占用空间小的救援机构，选择将机构的水平移动改为旋转，放弃了

通过解除单向门限位这一方式的释放，转而通过直接抬起机构完成释放。虽然释放不如往届救援机构，但保留了单向门抓取容易的优点，同时节省了空间，并且增加机构寿命以及减小维护难度。



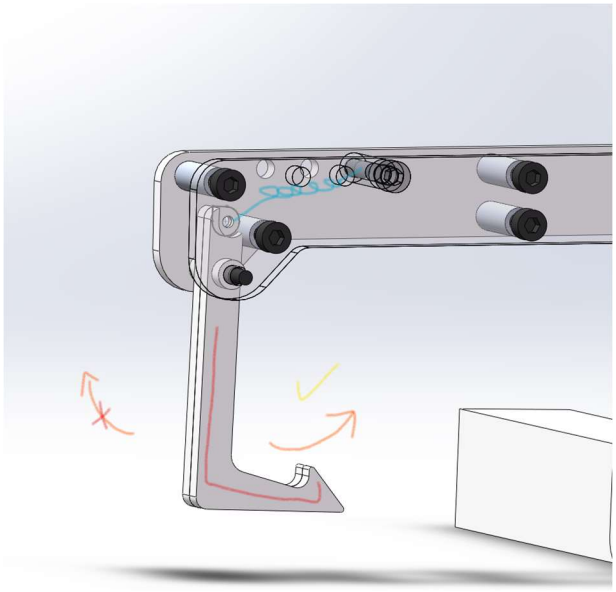
改进后的救援机构



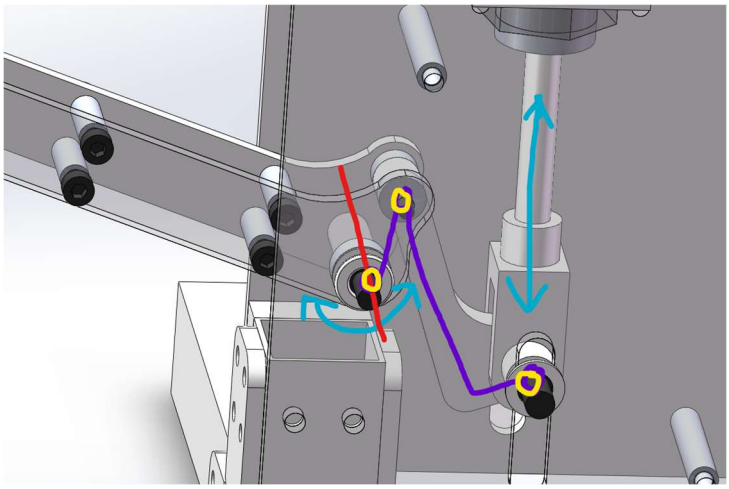
释放与收起

救援机构主要由：钩（黄色），臂（红色），连杆（紫色），气缸（青色）组成。

钩：单向开合，通过拉簧（蓝色）复位。



连杆机构：将气缸水平运动转为臂绕轴旋转。

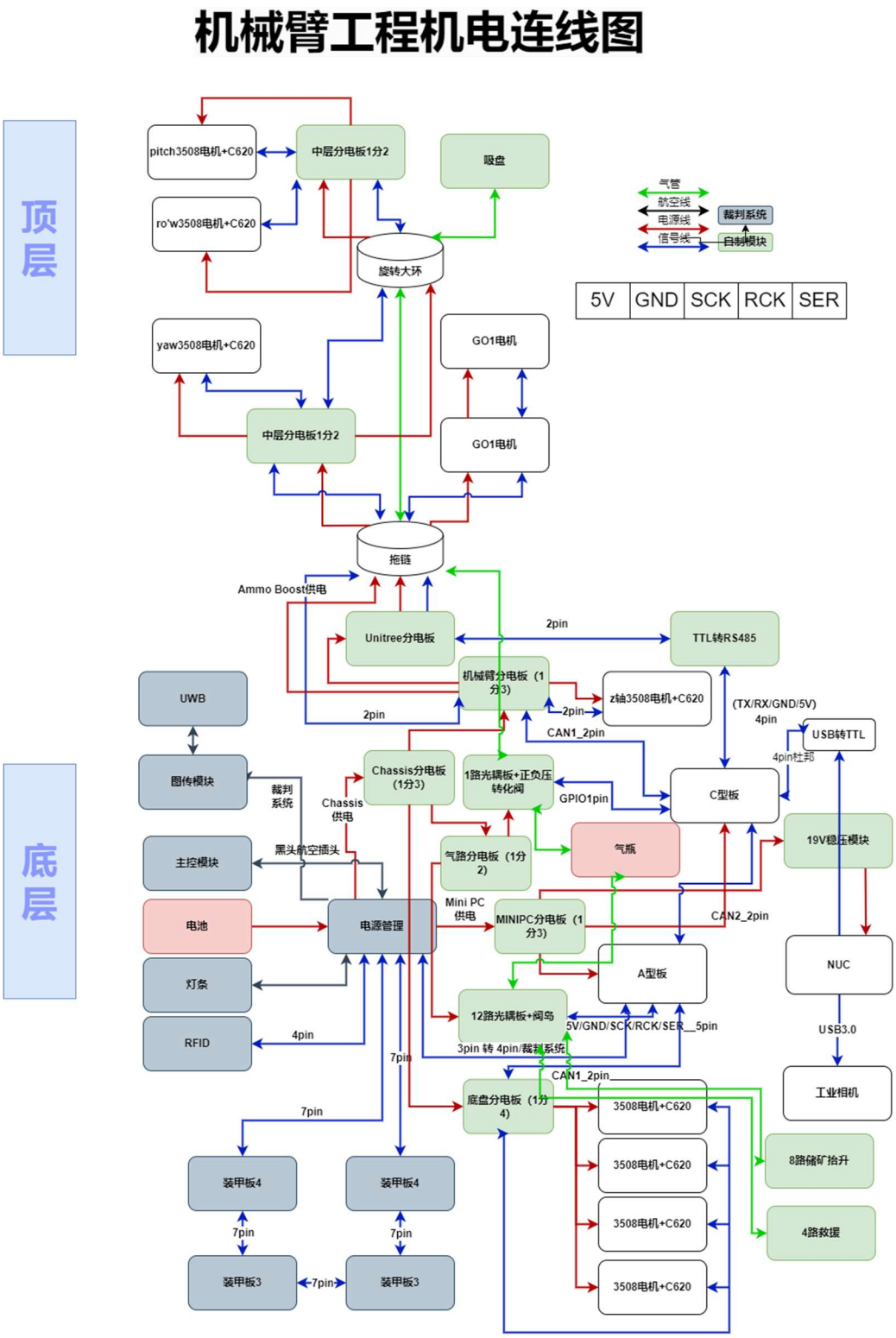


臂的限位与固定：通过槽口限制臂转动角度。

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：

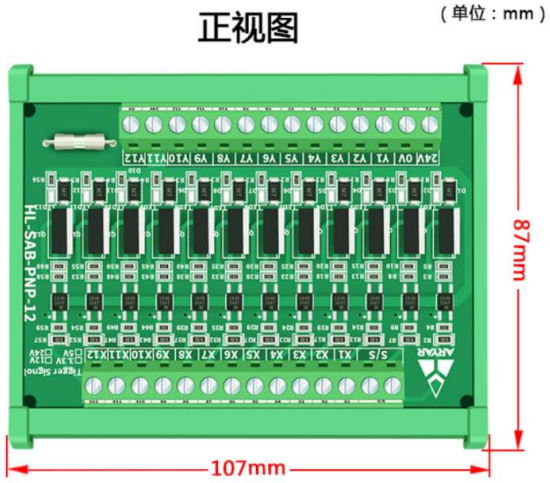


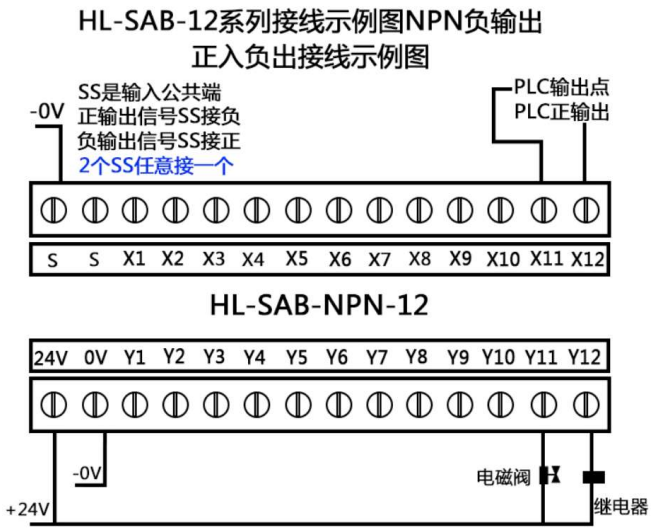
气路自研控制板

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

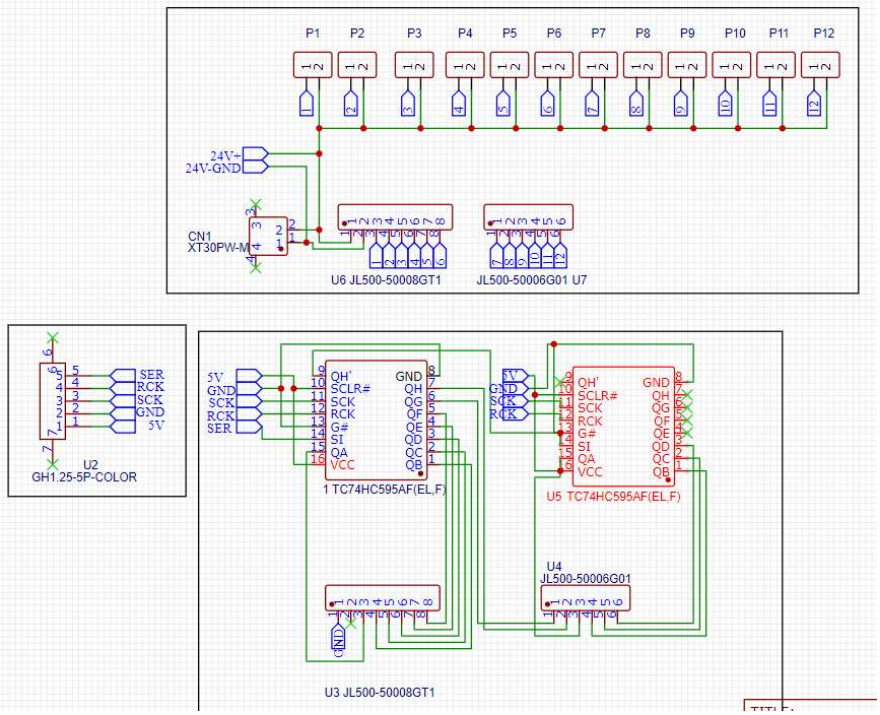


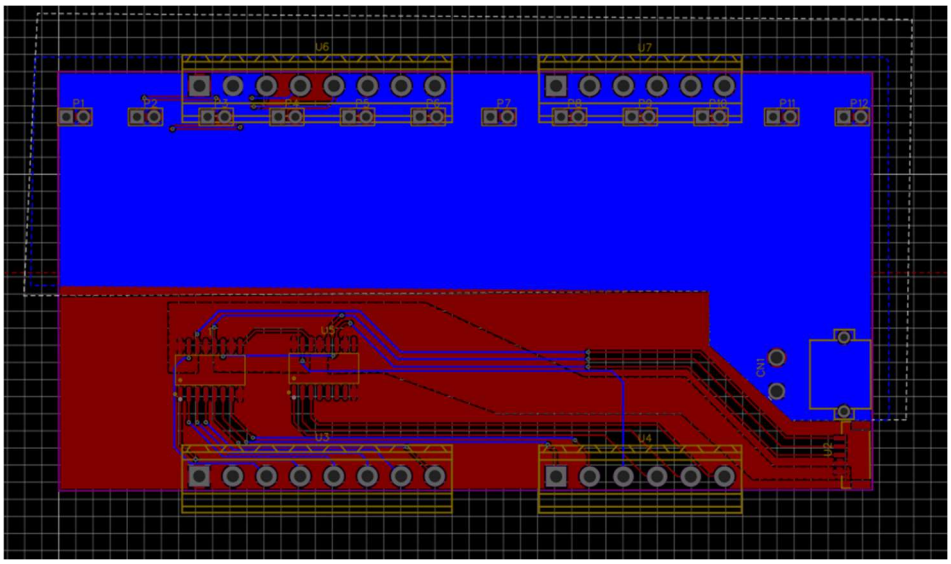


光耦板采用 HL-SAB-NPN-12 和 HL-SAB-NPN-1 型号，其设计均为共阳极的负极输出，方便与气阀连接进行控制。其核心目的在于处理驱动问题，使得控制电路和驱动电路隔离开来，每个 X 输入端对应相同数字的 Y 输出端，当位移寄存器输入高电平，光耦板输出端将输出-24V 电压，在位移寄存器所在电路板上形成 24V 的输出使气路电磁阀导通。

位移寄存器

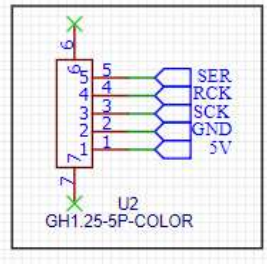
位移寄存器控制板的原理图和 PCB 设计图如下图所示：





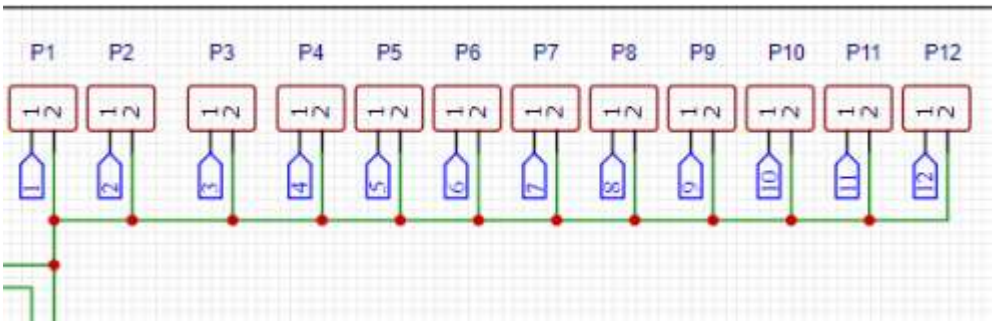
我们可以将其抽象成几个部分：

1. 输入部分：



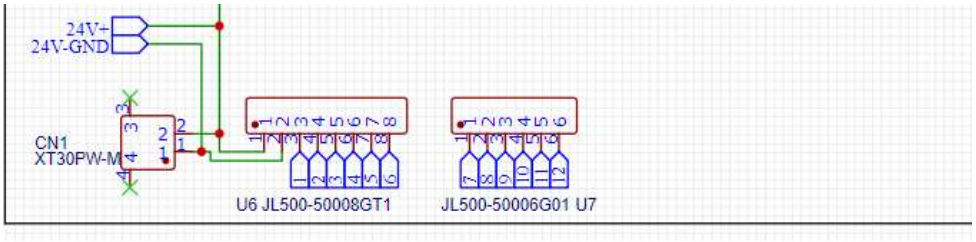
此处为 5pin 卧贴，每一路连接位移寄存器对应的端口。

2.输出部分：

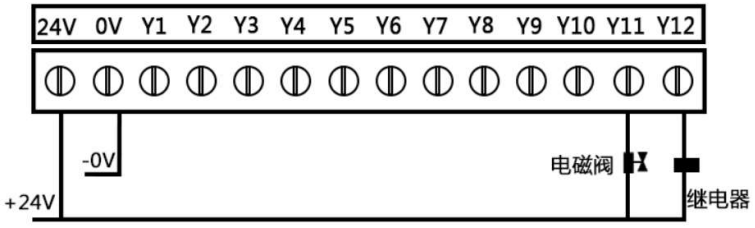


上图 P1 至 P12 分别连接至气阀上，1 至 12 端口为负输出口，另一端为共阳极。

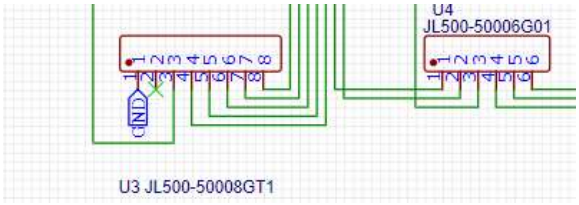
3.光耦板输出：



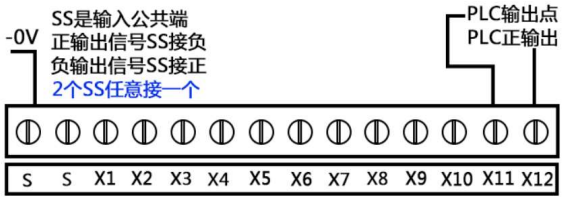
CN1XT30PW-M 为 XT30 卧贴，提供 24V 电源。U6 JL500-50008GT 和 JL500-50006G01 U7 为电路板上焊接光耦板输出口的部分，对应光耦板接线示例图如下部分：



4.光耦板输入：

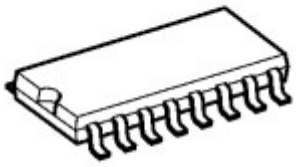


对应光耦板的输入端的焊接部分，对应光耦板接线示例图如下部分：

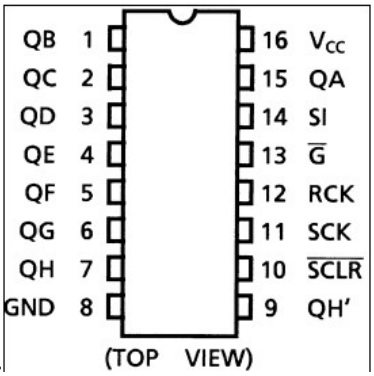


5.寄存器部分：

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



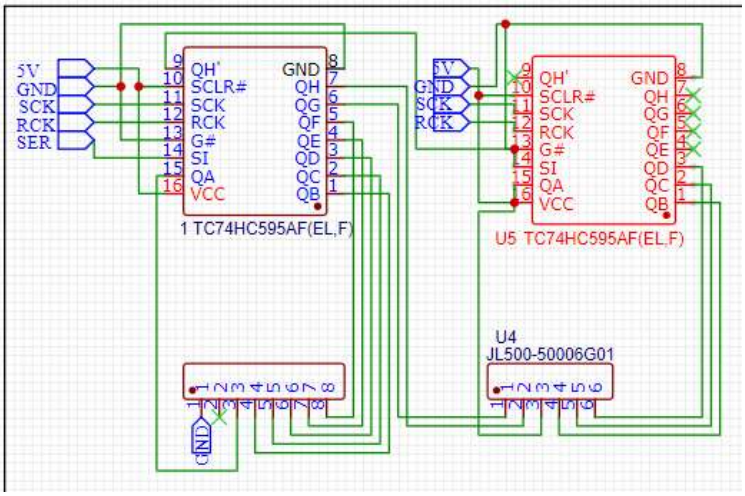
位移寄存器控制逻辑真值表如下：

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();
    }
```

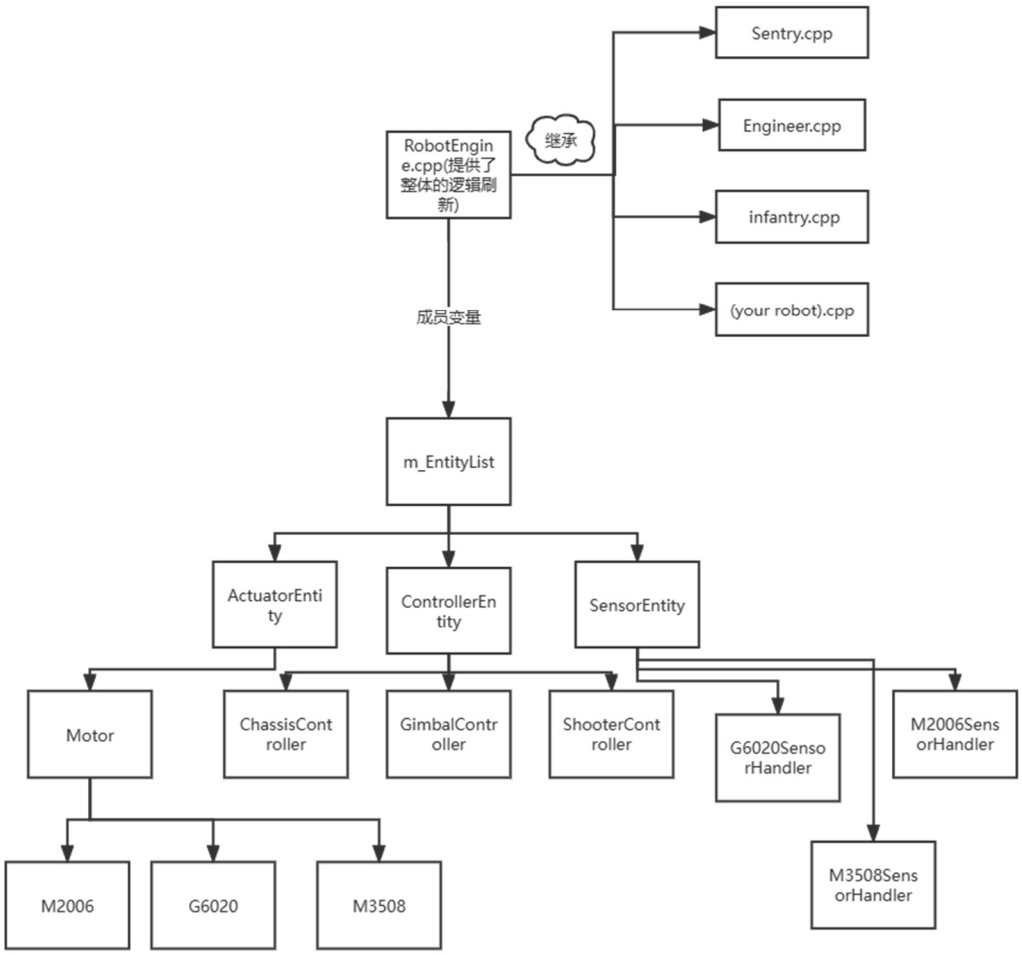
机器人更新

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();
    // ...
}
```

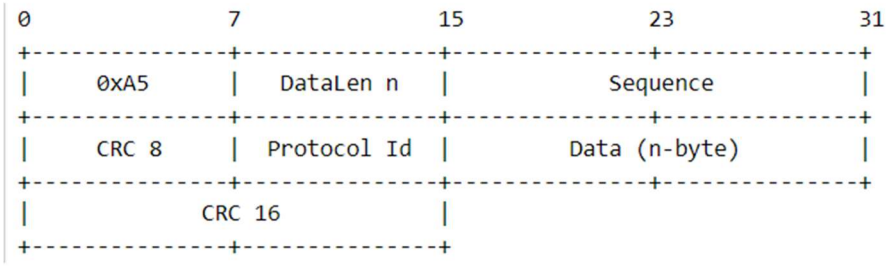
RobotEngine::Tick()，会遍历所有记录下来的 Entity，检查是否应该在当前执行其 Update()成员函数。Tick()应该每个 Systick 都调用一次。有三种类型的 Entity，分别是 SensorEntity，ControllerEntity 和 ActuatorEntity，每次调用 Tick()时，只有在所有 SensorEntity 都遍历了之后才会开始遍历 ControllerEntity，同样，遍历完所有 ControllerEntity 才会遍历 ActuatorEntity。相同类型的 Entity 之间执行顺序不确定。

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

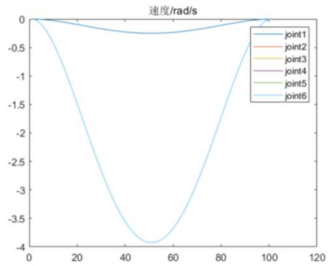
鉴于我们设计的类 SCARA 构型的机械臂，我们的驱动器将不再负载整个末端负载的重量，关节驱动器只需提供瞬时加速所需的驱动力和对抗系统的动摩擦。这可以将我们的电机力矩最大利用到加速上，虽

然损失了一定的的稳定性但在速度上，我们有绝对的优势。

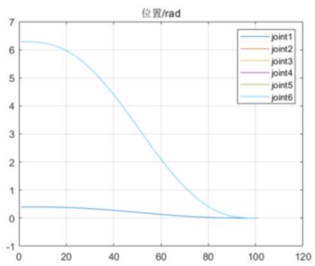
为了保证我们的运动性能，电机需要以一定的速度要求提供足够的扭矩。我们尝试对机械臂的工况进行动力学仿真，计算出电机所需要输出的功率。

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。

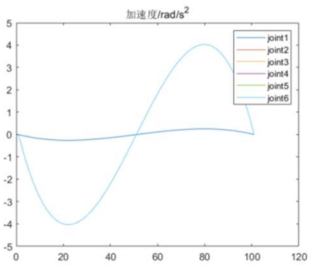
我们的工况选择了将每个关节一整周，并在末端负载 4kg，模拟矿石和末端执行器，整个流程在 3s 内完成，使用线性 5 次插值计算的加速度，速度和位置规划如下：



关节速度图



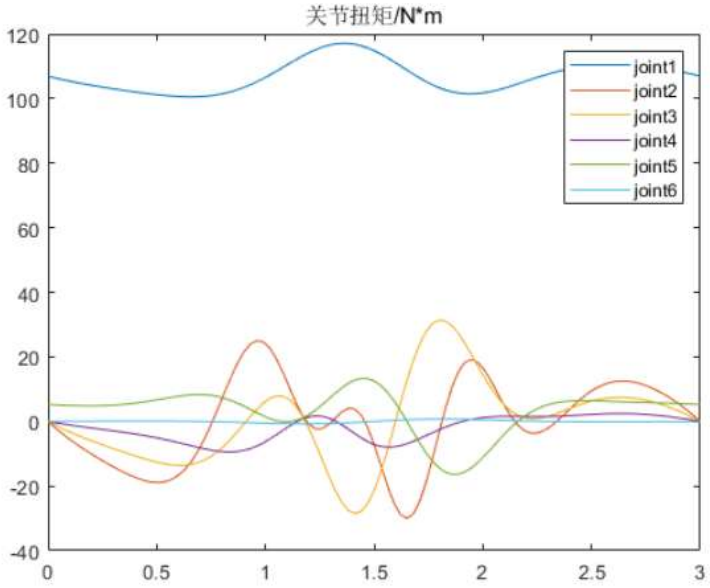
关节位置图



关节加速度图

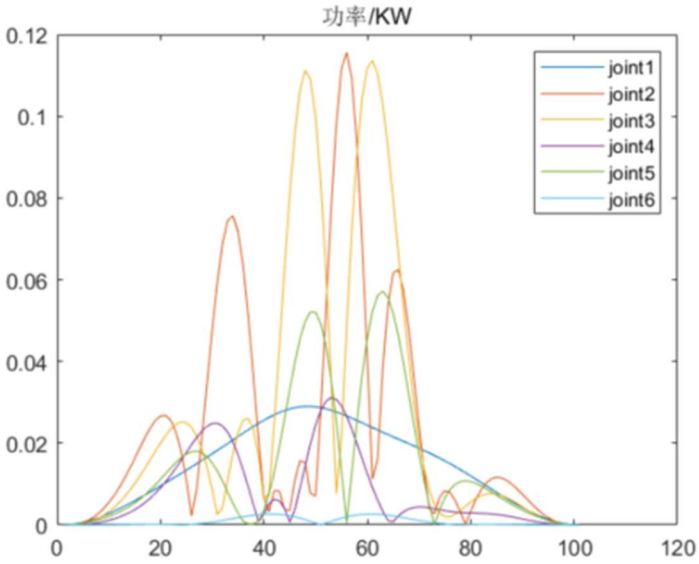
由于我们的机械臂需要做到连续的坐标变化，所以我们关注于需调速的机械。选择几个关键字：负载，短期运行，断续运行，调速平滑程度高，无特殊场所，变功率负载特性。根据机械设计手册可得：**调速范围 1：3 以上，需连续稳定平滑调速的机械，采用直流电动机。**

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。

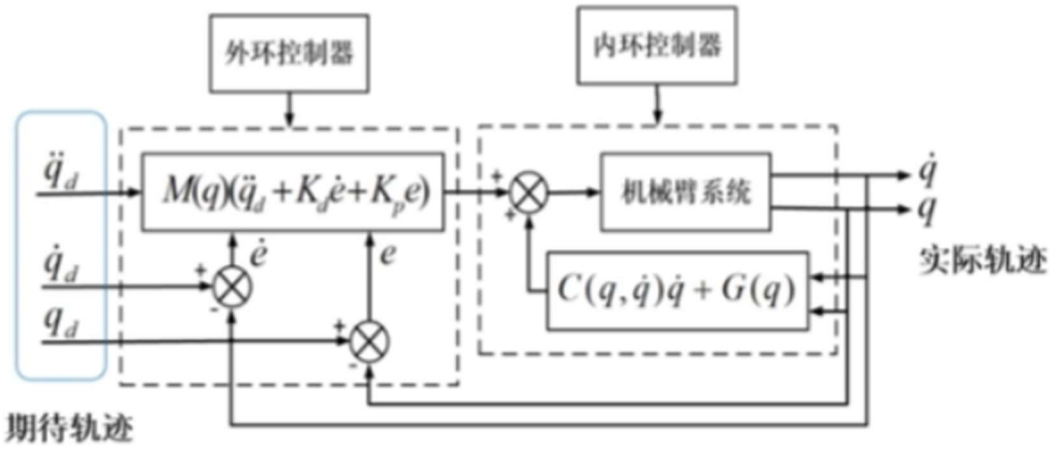
我们选用的链轮为 2 分 10 齿的链轮，传动半径为 10mm，额定扭矩为 3NM 的电机能输出 300N 的力，大于 T_N1，满足需求。

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：

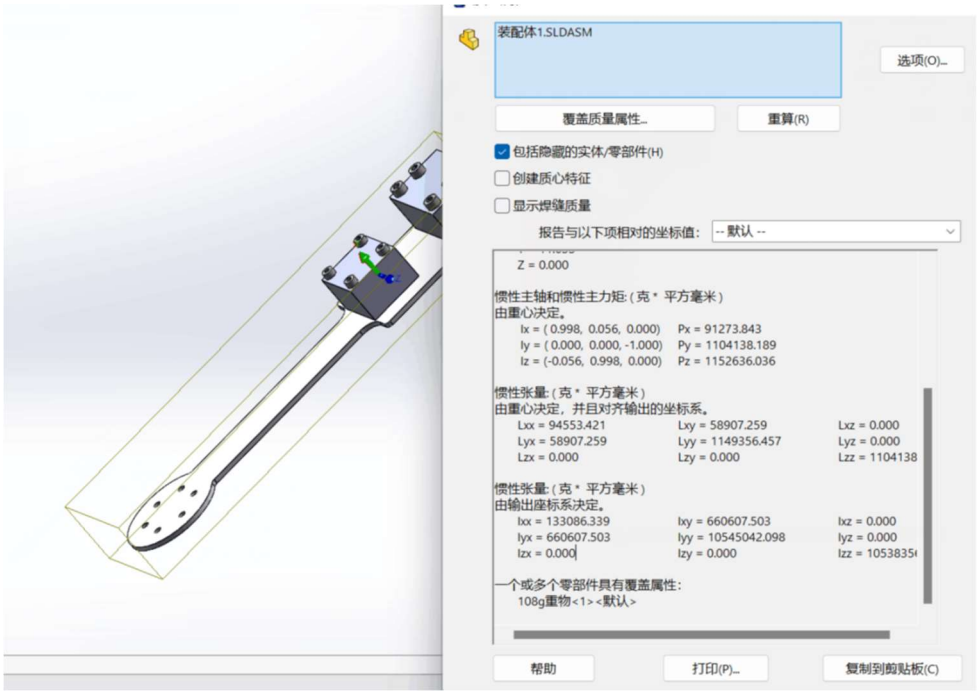


前馈力矩在整个控制系统的输入中占大部分，它的得出是通过解耦机械臂每个关节的运动。这种**非线性补偿器**能够很好的解决单负反馈解决不了的**奇点力矩突变**等问题，而反馈力矩只做补偿作用，在控制系统的输入部分只占小部分。

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**

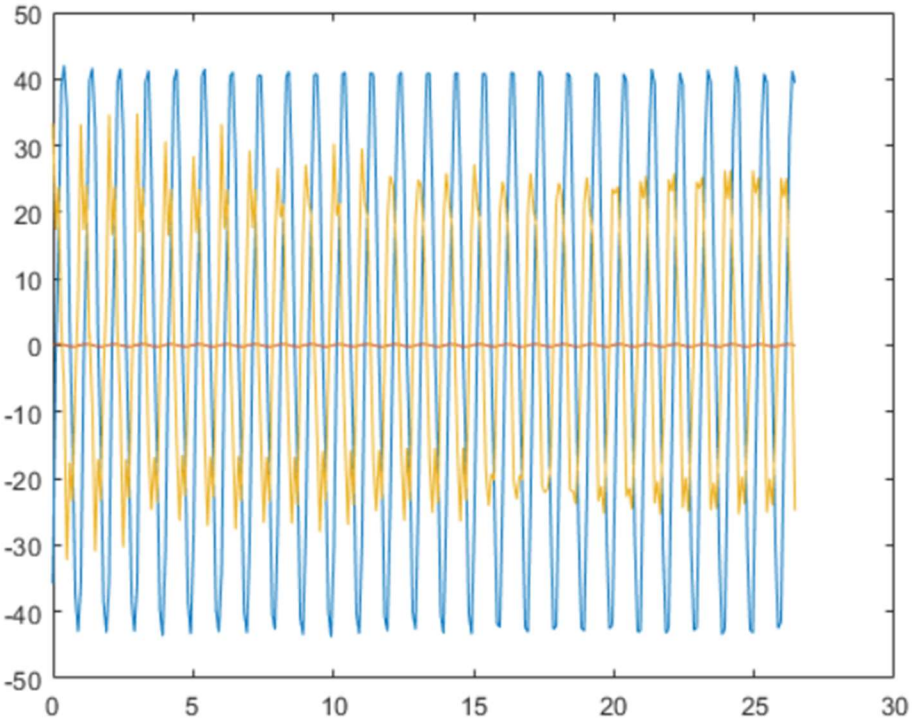
根据 Solidworks 的建模我们得出了该负载的转动惯量如下图所示：



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:

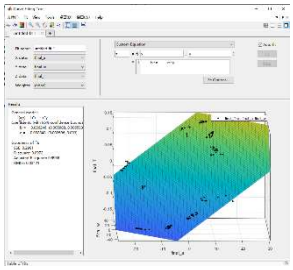
$$T = k_1 \times a + k_2 \times w$$

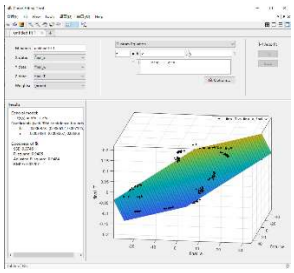
我们采集的数据如下图所示：

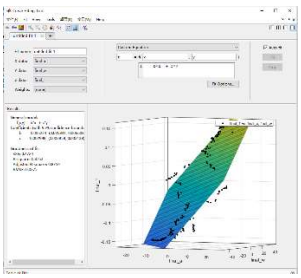


我们总共做了 40 次实验，每个条件 10 次，最后取平均值

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

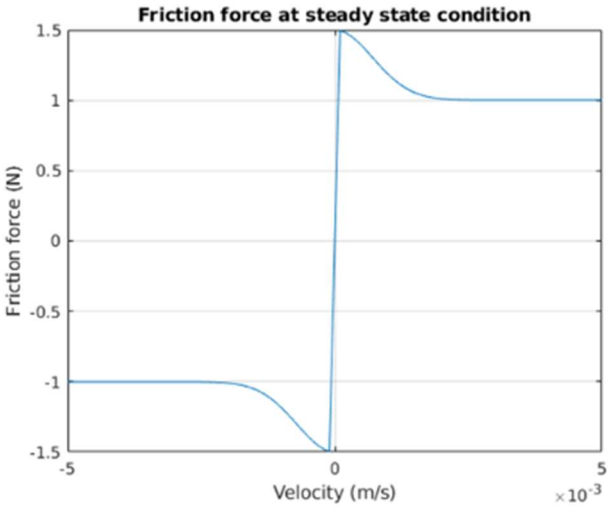
$$M = \alpha * 1$$

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。

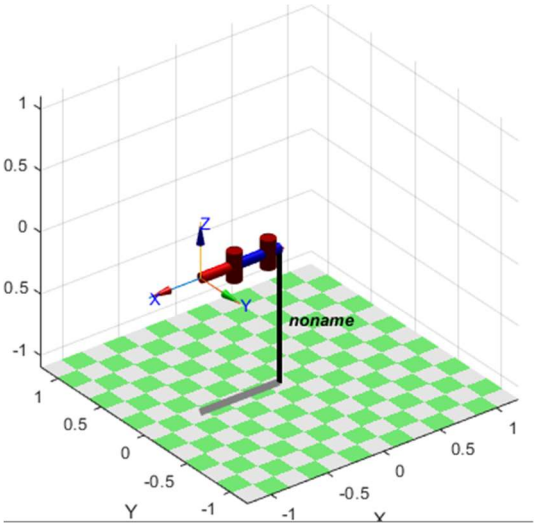
最后所有的关节都以相对平稳的方式运动。

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。

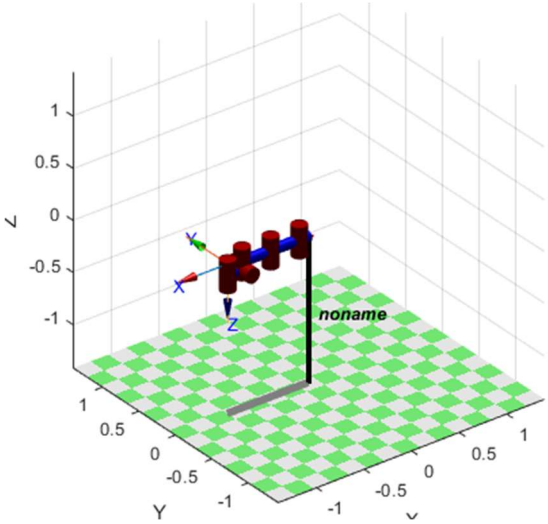
我们希望的是做到运动学解耦，位置三轴和姿态三轴隔离开。

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：

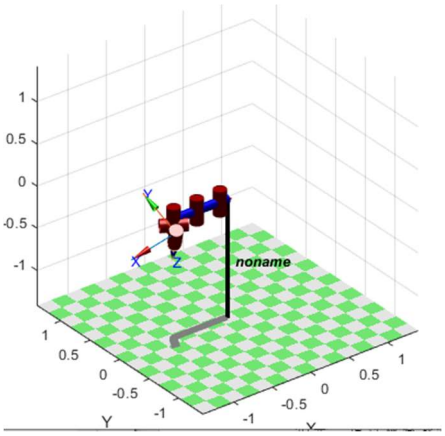


整体机械臂型图：PRR+RRR

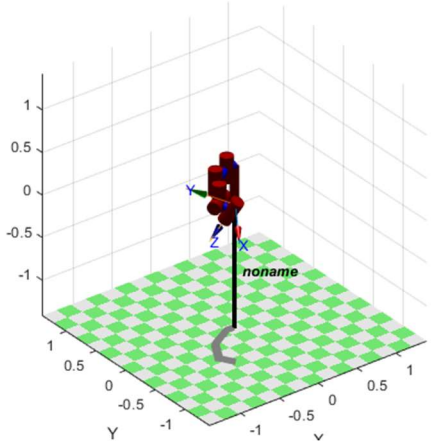
那么无论第 2 关节第 3 关节怎么旋转，第 4 关节的旋转轴都不会改变，因为们的旋转轴平行，如果不平行，那么我们提供调整补偿作用的关节的旋转轴就会偏转失去了补偿的能力。

具体解算流程如下：

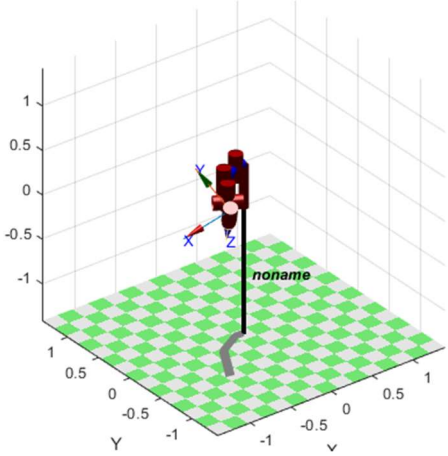
1. 根据所需要的姿态角反解出第 456 关节的角度。
2. 根据第 456 关节的角度，计算出位置偏移
3. 根据所需要的位置减去位置偏移得到关节 23 旋转的角度和关节 1 的伸长高度。
4. 最后旋转第 4 关节补偿由于位置三轴旋转带来的偏移。



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：

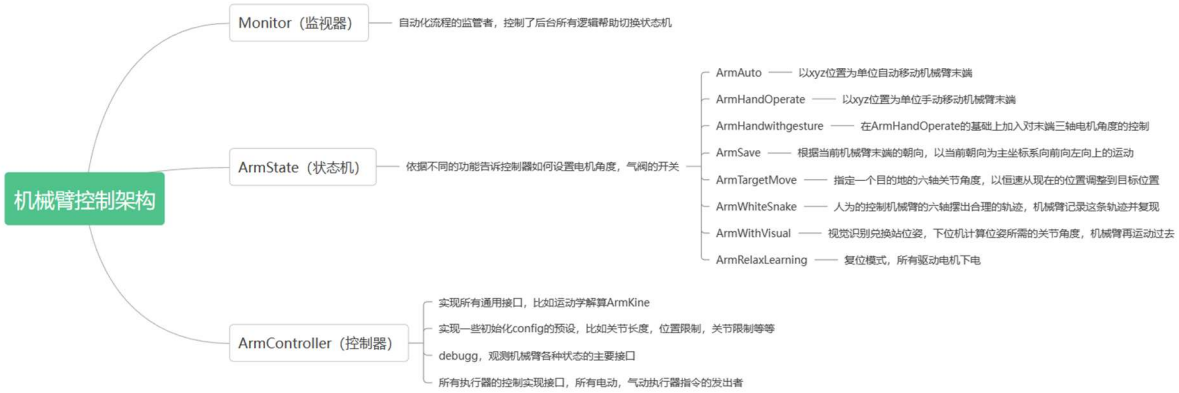
- 如何控制电机回到初始设定位置（本机械臂所有的电机均不含绝对编码器）？

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 =

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

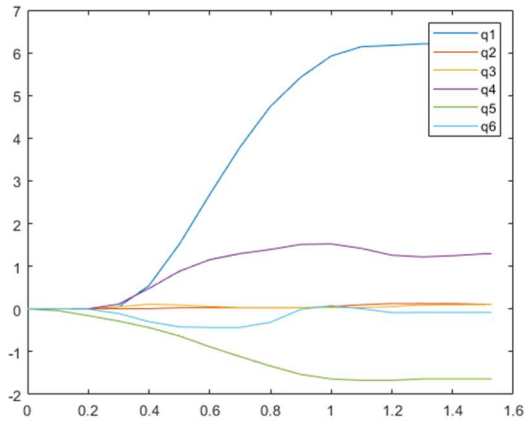
$$\sigma_1 = \cos(\theta_4)\sigma_3 - \sin(\theta_4)\sigma_4$$
$$\sigma_2 = \cos(\theta_4)\sigma_4 + \sin(\theta_4)\sigma_3$$
$$\sigma_3 = \cos(\theta_2)\cos(\theta_3) - \sin(\theta_2)\sin(\theta_3)$$
$$\sigma_4 = \cos(\theta_2)\sin(\theta_3) + \cos(\theta_3)\sin(\theta_2)$$

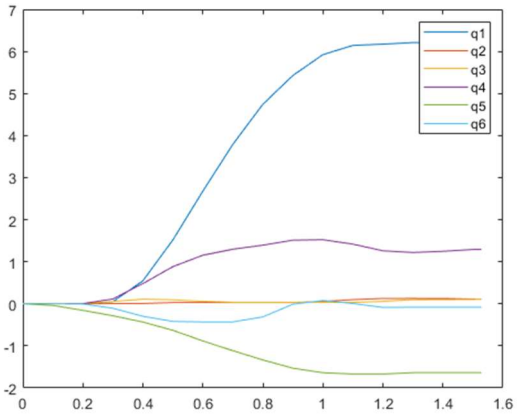
于是我们需要找到这是的 z 轴方向，并使用这时的 z 轴在世界坐标系下的 xyz3 轴的分量来进行控制：

```
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：

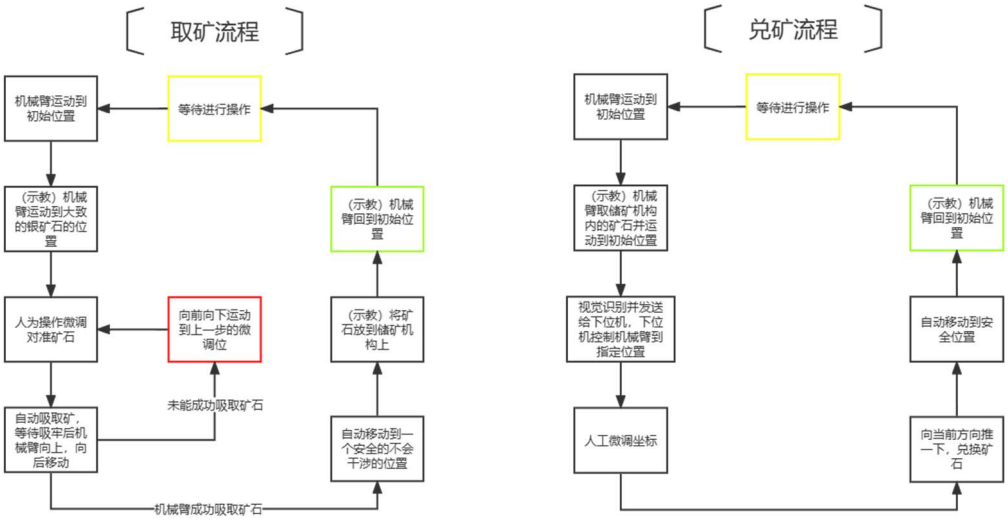




可见效果十分优秀。

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。

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

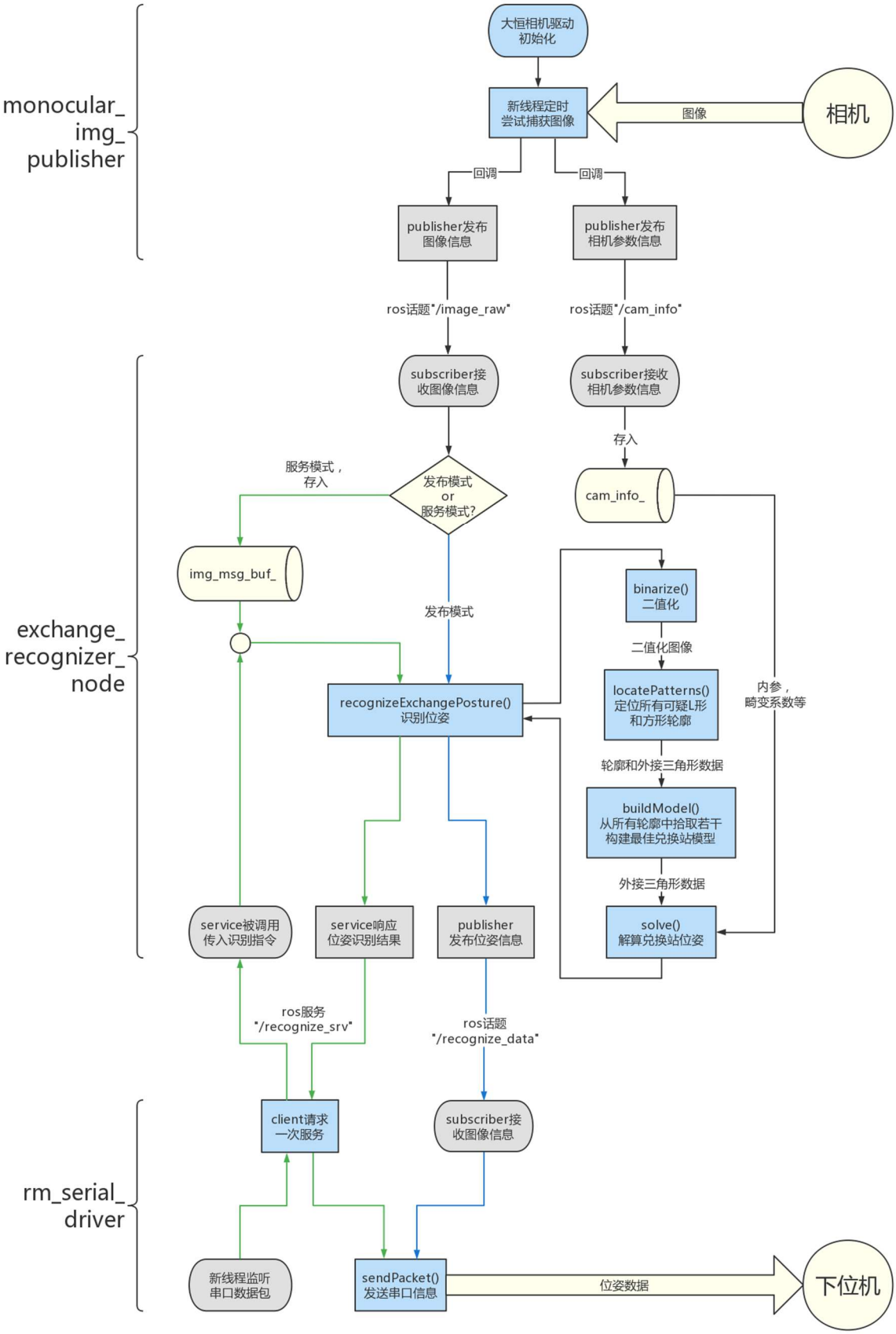
运行环境：NUC 一台，需安装 Ubuntu 20.04 系统，Eigen3 线性代数库，ros2 系统(foxy 版本)，大恒图像相机 SDK，以及 ros2 的 serial_driver 和 image_transport 插件。

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

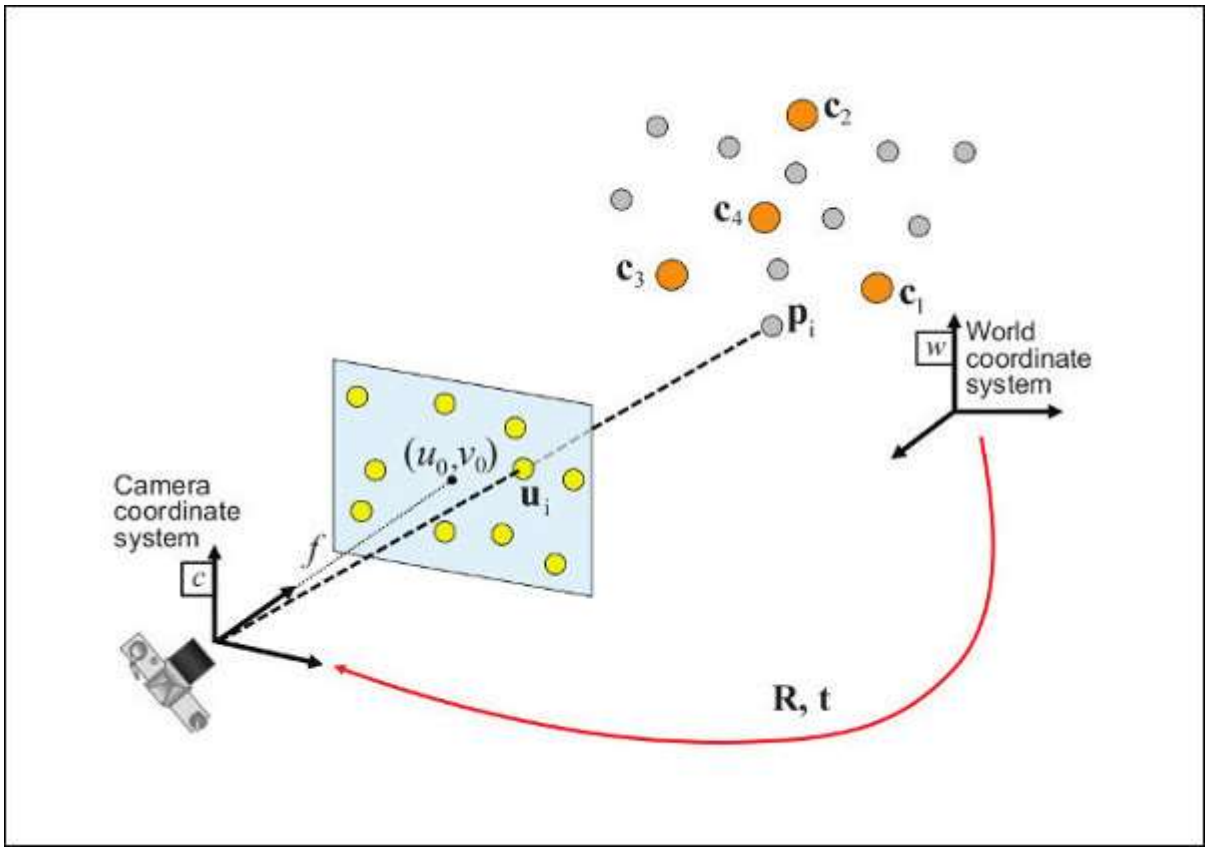
将三个 L 形图块按顺序组合即为一个兑换站模型，不考虑方形图块。将第一、第二个 L 形图块的不对齐度与第二、第三个 L 形图块的不对齐度相加作为总误差，使用剪枝 dfs 来从所有 L 形图块中寻找误差最低的组合模型。

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

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

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

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

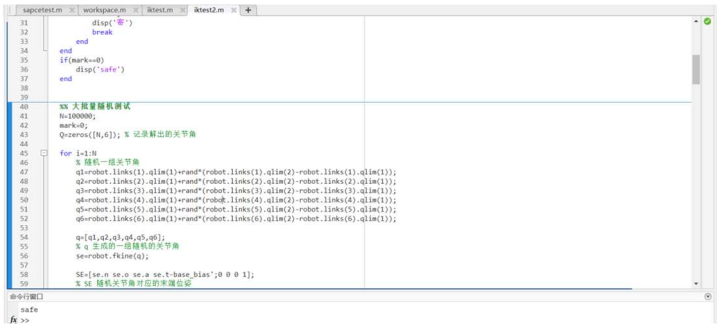
我们需要验证我们的机械臂是否一定能达到兑换站目标空间中的所有位姿（上面的工作空间只能看出位置，不能看出姿态），因此，我们需要进行兑换站所有位姿的可达性测试。

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

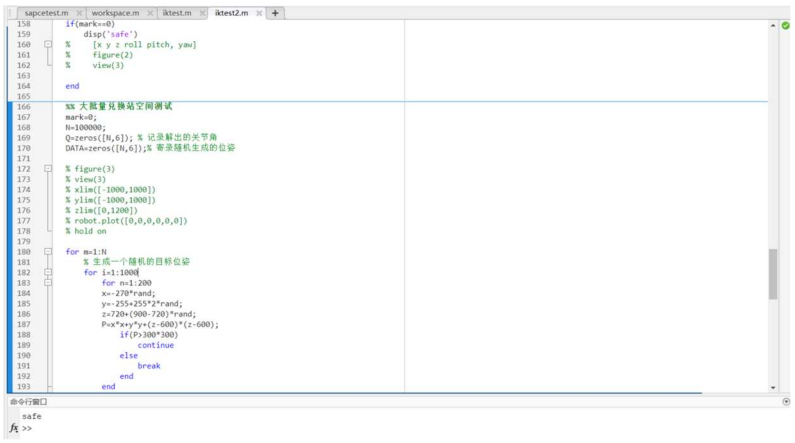
接下来，我们随机在官方规则给出的兑换站运动空间/角度范围内，生成一系列兑换站位姿，然后用我们的解算去解出相应关节角，通过大批量随机数据的检验，来确定我们的构型一定能满足规则的需求。

```
%% 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')
```

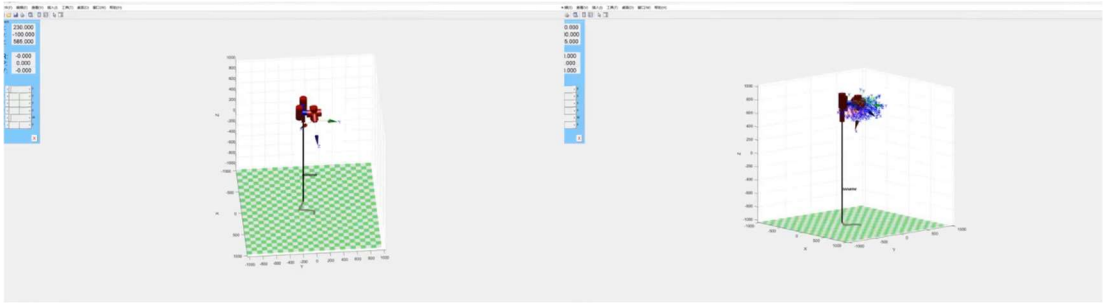
```
        break
    end
end
if(mark==0)
    disp('safe')
%     [x y z roll pitch, yaw]
%     figure(2)
%     view(3)

    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）

不同等级兑换测试

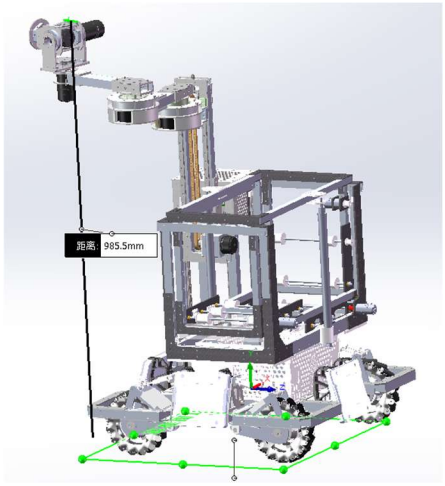
测试目标：测试机械臂完成 0-4 级兑换的能力

测试流程：0-4 级兑换各随机摆 5 个位姿，尝试利用机械臂完成兑换

测试结果：能完成 0-3 级的全部位姿，4 级的部分位姿可能达不到

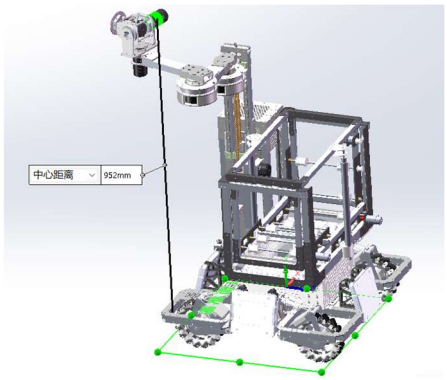
测试结果分析：

我们在确定机器人最大伸展尺寸（高），希望机器人的最高尺寸为 985mm，这个高度指的是机械臂第 5 关节的某个结构的最高点到地面的距离。如下图所示：



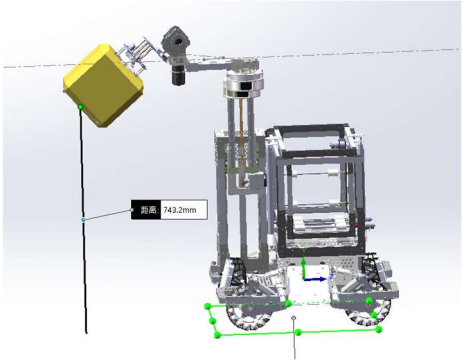
到这个电机固定板的距离为 985.5mm

实际上，第 5 关节的轴线离的最高高度为 952mm。



第五轴的高度实际要低一点

当 4 级兑换的 pitch 角为 60° 时，因为有第 6 关节连杆和矿石本身长度的叠加影响，使得实际的矿石最外表面中心点的高度比这个高度低很多。如图，在这种极端情况下距离为 743.2mm，这代表 4 级兑换时出现的 pitch 角的大小会导致我们能兑换的最高高度（z'）下降，因此 4 级兑换的部分情况我们不能到达。（反思了一下，当时做仿真时只考虑了解算的完备性，没有考虑超高等条件，导致了悲剧的发生）



Pitch=60°时的极限兑换高度（z’）

4 级兑换补充测试

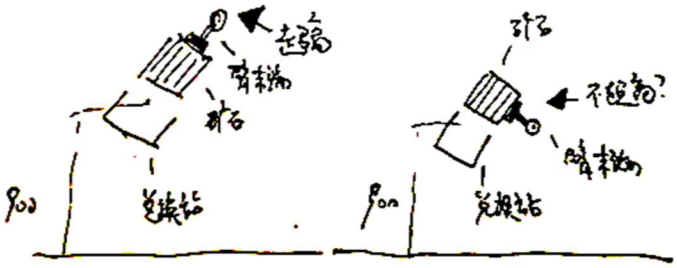
背景：书接上回，我们发现现有的构型和尺寸不能应对 pitch 和 z 轴都较大的 4 级兑换情况，在此基础上，我们思考了在现有基础上补救的方法。

如图所示，我们猜想，对于 pitch 角度较大的情况，我们可以试图改变机械臂拿矿的方向，这样就不会超过高度限制。可能遇到的问题有二：

1. 矿石方向会发生偏转
2. 因为摩擦力，矿石下不去

对第一个问题，根据我们之前去场地时的测试，翻转 90° 的矿石是有很大可能被识别到的。

对于第二个问题，我们对此展开测试，试图测量能滑下去的最小 pitch 角度。



新方案示意图

测试目标：在改变矿石持有方式（如上右图）的情况下，测量能滑下去的最小 pitch 角度。

测试流程：从 0-60 改变兑换站的 pitch 角，用机械臂尝试兑换，并记录 pitch 角度和是否滑下去。

测试结果：在 pitch 角大于 35° 时，可以滑下去。

基于这个测试结果，我们后续准备改变机械臂取矿的运动规划。但是由于我们的兑换站实际上摩擦系数等可能和官方的不一样，所以到场上进一步测试。

取矿测试

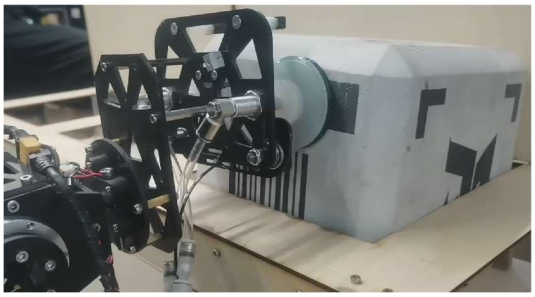
测试目标：用机械臂取到小资源岛的矿石。

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

测试结果：成功率为 0。

结果分析：

从侧面取矿时，由于吸盘需要一定正压力才能吸紧，在吸紧后由于侧向压力的原因与小资源岛挡板发生摩擦自锁，导致矿石不能被拿起。



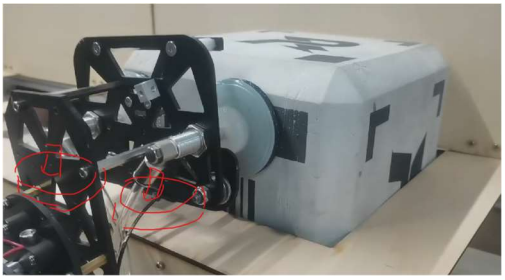
矿石在拿起过程中变倾斜，发生自锁

我们尝试了不同角度以及尝试机械臂调整位置和姿态，发现还是很难拿出矿石。

取矿补充测试

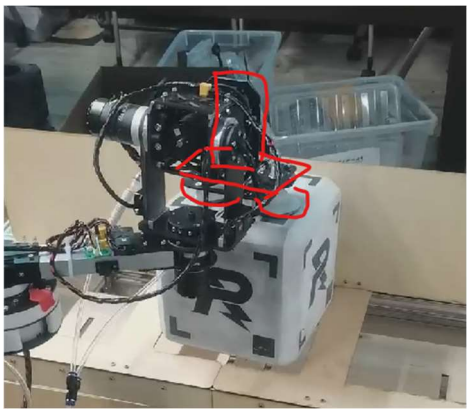
背景：书接上回，我们发现侧面拿矿石的设想不能实现。但是机械臂的构型牵一发而动全身，有没有与现有方案解耦的方法能解决取矿的问题？

于是我们设想，在原有末端执行器两个吸盘的下方再额外增加两个吸盘，这样能够从上方拿出矿石，也不影响原来的吸盘（原来的吸盘方向要用来储矿和兑换，不能删除）。



在下方再加两个吸盘，不改变原来的构型和吸盘

为了迅速的验证这个想法，我们改变了原来吸盘的安装方式，快速的进行了测试。



改变了安装方式，吸盘在下方

于是我们通过这个临时的机构来测试从上方取矿的成功率。

测试目标：测试从上方取矿的成功率。

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

测试结果：成功率 100%，从上方可以很轻松的取出矿石。

因此在原有机构下增加两个吸盘会作为我们下一版的方案。

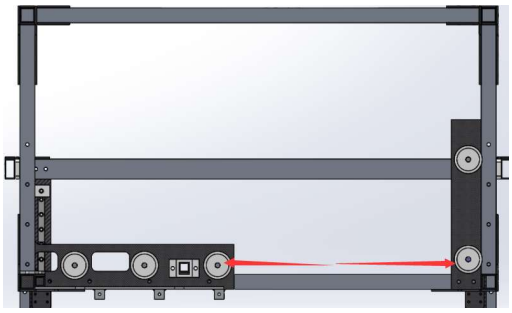
1.6.1.2 储矿机构

轮组间距测试

在样机中，我们设计可移动的轮组，可以修改轮组间的距离，测试矿石由上层进入下层时姿态变动的变化情况：

可左右移动的轮组

由螺丝固定两侧玻纤，原本孔处改为直槽口。在螺丝松时可移动。在螺丝紧时，延圆心连线方向力较小，尽可能不移动。此装置仅用于测试，在确定尺寸后不在出现，未考虑寿命问题。



初版样机

测试目标：改变轮组间距，尽可能实现矿石姿态稳定变化。

测试结果：

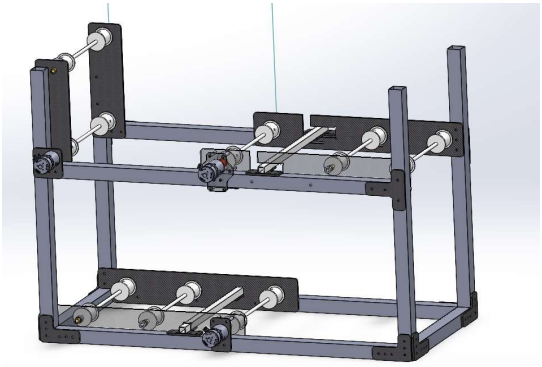
间距较近，且与矿石接触侧面轮组面向上移动时，矿石可实现不旋转落下，但失误率较高。会出现连续翻转情况。

间距较远时，矿石可实现旋转 90 下落，但第三四矿石下落时易卡住。

经改变电机转动方向，解决矿石卡住的问题。

在下落处卡住，矿石开始连续翻转

矿石与上层产生干涉



一代储矿机构

本次测试基于一代储矿机构进行测试。一代并未设计抬升机构，上层空间削去一半。在满足限高的同时完成矿物移动。

一代测试发现的问题：

1. 此时高为 400mm，仍有些过高，不满足需求
2. 考虑到赛场激烈对抗。部分大弹丸进入影响储矿运动
3. 轴平行度不良，影响转动及电机
4. 与机械臂联调时。机械臂自身存在机械最低限位，最低无法从储矿机构中取矿，储矿机构需整体抬高，或在储矿底加一级抬升，无空间完成新抬升。
5. 四周对矿石限位不足，矿石可左右移动。
6. 层高较低，矿石在下层时易与上层同步带干涉。

1.6.1.3 末端执行器

真空发生器	箱式 10		
吸盘	80mm 风琴+薄海绵	**吸盘**	100mm 平+厚海绵
气源表压	**真空度 kPa**	**气源表压**	**真空度 kPa**
0.05	-	**0.05**	11

0.1	-	**0.1**	21
0.15	-	**0.15**	24
0.2	39	**0.2**	36
0.25	50	**0.25**	50
0.3	63	**0.3**	64
0.35	84	**0.35**	83
0.4	95	**0.4**	94

真空发生器在 0.3L 20MPa 供气时使用时间对比

真空发生器	管式	07S			
吸盘	80mm	风琴+薄海绵	**吸盘**	100mm	平+厚海绵
气源表压	**真空度 kPa**	**吸附情况**	**气源表压**	**真空度 kPa**	**吸附情况**
0.05	-		**0.05**	-	不行
0.1	-		**0.1**	-	不行
0.15	-		**0.15**	14	勉强
0.2	24		**0.2**	21	
0.25	36		**0.25**	33-35	
0.3	54		**0.3**	47	
0.35	72		**0.35**	62	
0.4	86		**0.4**	83	
0.45	89		**0.45**	86	
0.5	89		**0.5**	86	

相同供气气压下吸盘对真空度的影响

气源表压 MPa	真空度 kPa	吸附情况

0.15	18	掰掉很轻松，但是可以承受碰撞
0.2	25	
0.25	33	可以稳固吸取矿石
0.3	41	
0.35	50	
0.4	58	
0.45	65	

供气压力测试

1.6.1.4 救援机构

基本功能测试

测试目标：测试抓取和释放等功能。

测试结果：抓取和释放能在 1s 内完成，并且操作手操作反馈比较顺畅；但是左右平移时可能出现脱钩现象。

左正常钩取，右脱钩

刚度测试

测试目标：对救援中的机器人进行旋转和撞击，检验机构刚度。

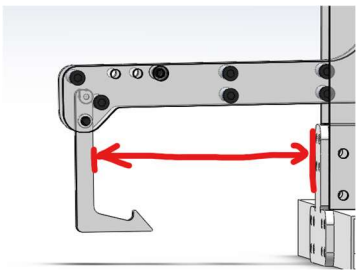
测试结果：机构未出现明显失效。

左旋转，右撞击测试

一代救援机构总结

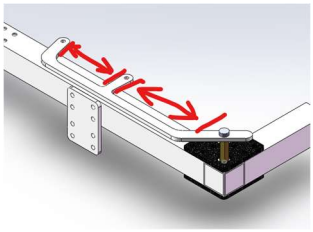
通过测试，发现以下问题：

1. 臂结构长度太长，在勾住被救援车辆后，两车间会有较大相对运动，影响救援，并增加脱钩可能。过长结构占用过多空间。



这里距离太长

2. 被救援部分设计间隔太大，在拖拽过程中被救援车辆可活动范围较大，容易脱钩。



被救援杆

脱钩

3. 臂收起时，钩无法折叠，占用空间。

钩露出无法收回

1.6.1.5 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误差很离异	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	0.010312	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	214.0	480.0	134.0	107.0	-0.007280	-0.033826	0.004805	0.999044	470.9	-11.7	32.0	成功	501.6	-11.6	29.6	0.015886	-0.016863	-0.035216	0.999111	30.75	0.08	-2.43	0.011394	0.016963	-0.040021	0.000067	-0.011407	-0.016953	0.945256	0.178968
中+平视 Roll45 重新校准yaw（	0.0	300.0	214.0	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	214.0	485.0	334.0	363.0	0.361166	0.368658	-0.248255	0.819768	617.6	9.6	276.3	成功	649.7	-2.6	362.98	0.377203	0.397857	-0.242300	0.800449	32.15	-12.18	-13.31	0.016037	0.029199	0.005955	-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	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
储矿机构 V1.0	功能： 实现无干扰情况下的 4 个矿石储存，并确定好关键尺寸。	2023.1.20
储矿机构 V2.0	功能： 实现 4 个矿石储存，加上外围挡板保护； 增加两级抬升机构，机构可两级收缩/伸展。	2023.4.10
救援机构 V1.0	功能： 实现撞击救援，基本功能正常。	2022.12.22
姿态识别算法 V1.0	功能： 实现位姿识别； 完成 NUC 部署。	2023.4.1
末端执行器 V1.0	功能： 实现吸盘的支撑，能够吸取矿石。	2022.12.28
末端执行器 V2.0	功能：	2023.3.2

版本号或阶段	功能或性能详细说明	完成时间
	实现吸盘的支撑，能够吸取矿石； 增加偏置，能给机械臂提供更大的 pitch 角； 增加气路保护。	
工程机器人 V1.0	功能： 兑换 0-3 级矿石，以及部分 4 级情况的兑换； 储存 4 个矿石； 全向移动。	2023.4.10

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	机械臂 V1.0	机械工师：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	末端执行器 V2.x	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	储矿机构 V2.0	机械工程师：刘乐

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
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	救援机构 V2.0	机械工程师：张宸翰

1.7 团队成员贡献

贡献度的标准由团队自行确定，与机器人最终效果和质量整体成正相关即可。

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&机械结构设计负责人	负责工程机器人整体的功能规划和项目安排； 负责机械臂 1-3 关节和底盘的结构设计	20%
盛李杰	机器人工程、大三、软件&硬件开发负责人	负责整个机器人硬件和嵌入式开发	20%
刘家荣	暂无专业、大一、算法负责人	负责视觉识别位姿相关算法的开发和调试	15%
刘乐	暂无专业、大一、机械组队员	负责储矿机构设计和测试	12.5%
张宸翰	机器人工程、大二、机械组队员	负责救援机构的设计和测试； 负责储矿机构的部分设计	12.5%
林沛君	机器人工程、大三、机械组队员	负责末端执行器的设计和测试	10%
肖星辰	暂无专业、大一、机械组队员	负责机械臂 4-6 关节的结构设计	10%

1.8 参考文献

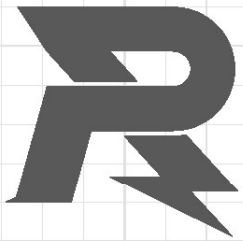
（实在没精力调格式了，谢罪）

《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



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