

GCR 系列协作机器人使用 手册（软件部分）

机型 GCR20-1100 & GCR14-1400

软件版本 2.2.14

UI 1.2.15



中科新松有限公司

版权所有©中科新松有限公司 2019。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制文档内容的部分或全部，并不得以任何形式传播。

商标声明

“SIASUN”、“新松”、“SIASUN 新松”等文字或形象均进行了商标注册保护，注册商标信息可见于公开的商标注册信息中。

本手册提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受中科新松有限公司商业合同和条款的约束，本手册中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围内。除非合同另有约定，中科新松有限公司对本手册内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本手册内容会不定期进行更新。除非另有约定，本手册仅作为使用指导，本手册中的所有陈述、信息和建议不构成任何明示或暗示的担保。

中科新松有限公司

地址：上海市浦东金桥出口加工区金藏路 257 号 邮编：201206

网址：<http://www.siasun-in.com>

目录

中科新松有限公司	1
1 简介	1
2 打开与关闭电源	2
3 界面介绍	5
3.1 移动界面	5
3.2 编程界面	11
3.3 运行界面	14
3.4 状态界面	16
3.5 设置界面	18
3.6 关于界面	29
3.7 断开界面	29
4 机器人脚本编程	31
4.1 创建程序	31
4.2 MOVE 标签：移动指令集合	34
4.3 FUNCTIONS 标签：函数集合	36
4.4 SOCKET 标签：网络通信指令集合	39
4.5 创建变量标签：	39
4.6 示例	40
5 机械臂外部通讯接口	43
6 脚本函数说明	51
6.1 简介	51
6.2 数据类型和变量	51
6.2.1 内置数据类型	51
6.2.2 系统常量	51
6.2.3 变量及变量的定义	52
6.3 表达式	53
6.3.1 算术运算表达式	53
6.3.2 关系运算表达式	53
6.3.3 逻辑运算表达式	54
6.3.4 赋值表达式	54
6.3.5 函数调用表达式	54
6.4 语句	54
6.4.1 循环语句 while	54
6.4.2 break 语句	54
6.4.3 函数定义	55
6.4.4 return 语句	56
6.4.5 条件控制语句	56
6.5 系统函数说明	57

6.5.1 移动控制.....	57
6.5.2 网络通信.....	60
6.5.3 系统函数及外设.....	63
6.5.4 调试相关.....	69
6.5.5 Modbus.....	70

1 简介

下面章节将介绍如何操作机器人。本说明书使用软件版本为 UI: 2.2.14

Robot:1.2.15，适配机器人为 GCR20-1100 型和 GCR14-1400 型协作机器人。

2 打开与关闭电源

主电源开关位于控制柜的接口面板上。接通 220VAC 输入并供电→按下系统开关→按下上电按钮→控制系统上电（开关指示灯亮起），等待示教器上的使用界面出现。



在欢迎界面点击“开始”按钮，

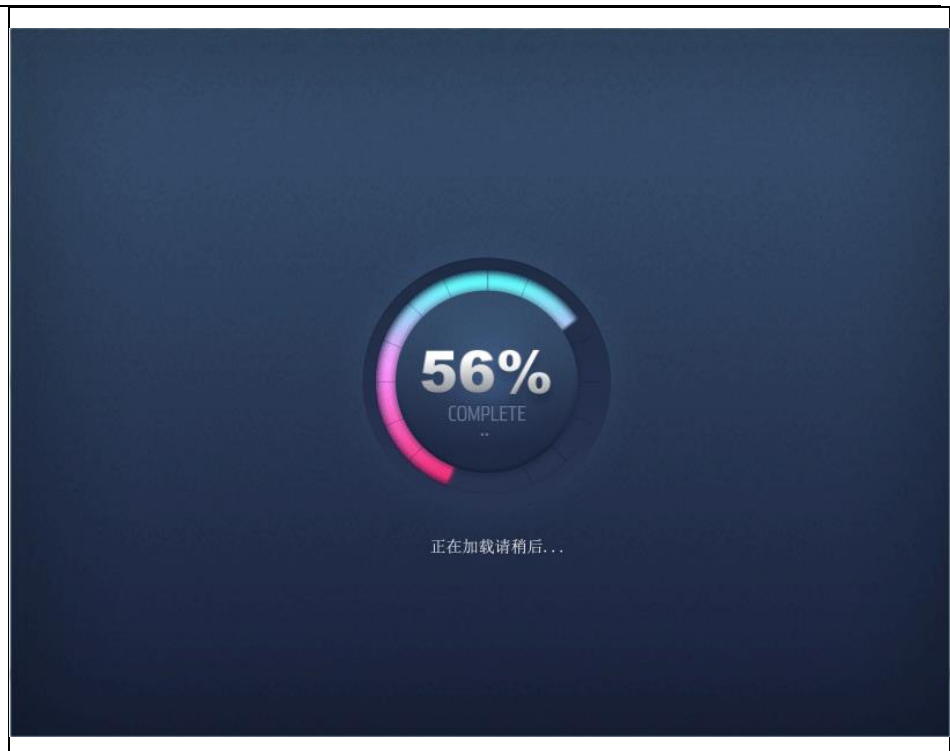


系统会显示机械臂安装方向及末端负载重量警告信息，如：“机械臂安装方向为：竖装 末端负载 0kg”。如果显示信息与实际情况不符，会对牵引示教和碰撞检测功能产生影响，请点击“修改安装方向”按钮进行修改。

确认安装方向和负载后，末端灯带亮白灯，表示机械臂已上电，初次上电需要等待机械臂各关节初始化，初始化完成后，关节右侧的红点会变为蓝色，此时可点击“全部使能”按钮开松开关节抱闸。



关节抱闸松开后，末端灯带会显示蓝色，示教器会进入到操作界面。

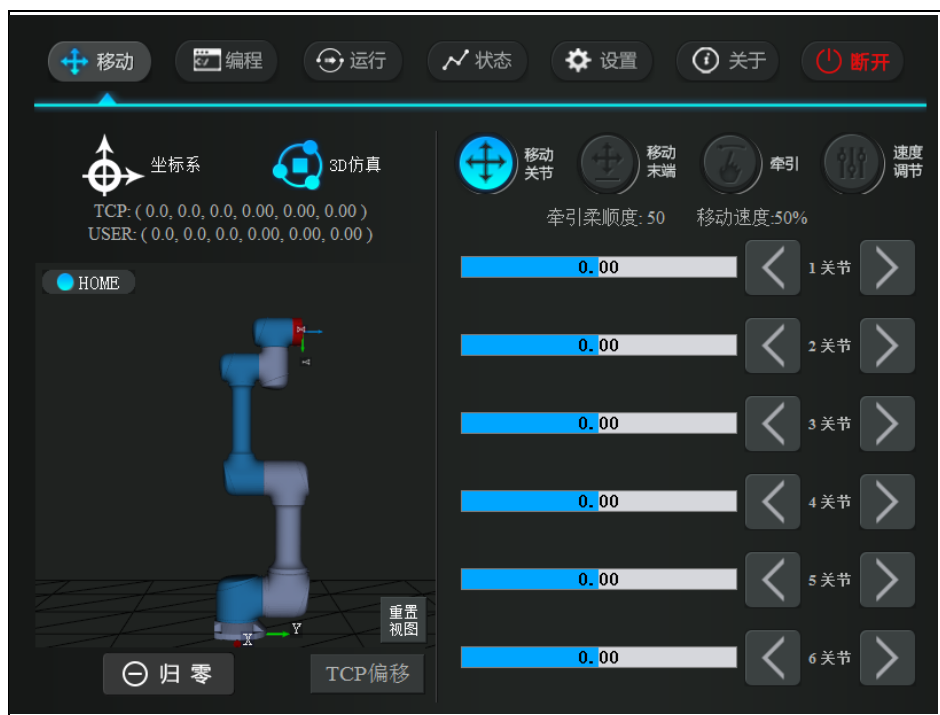


需要关机时，按下示教器界面“关机”按钮→机械臂断电→按下上电按钮断电→控制系统断电→断开 220VAC 输入。



3 界面介绍

操作界面共有 7 个子页面：移动、编程、运行、状态、设置、关于、断开。
可通过页面最上方的导航按钮来切换各个子页面。上电完毕后会默认进入移动界面。



3.1 移动界面

点击上方导航栏中的“移动”会进入移动界面。

移动界面中第二行有 6 个导航按钮：坐标系、3D 仿真、移动关节、移动末端、牵引、速度调节。同时会显示当前系统使用的 Tcp 参数以及工件坐标系参数。取消点选坐标系及 3D 仿真按钮，可切换到末端/关节位置输入界面。

1. 当前关节角度及末端坐标

左侧的数据显示了当前末端的位置以及各个关节的角度

点击数据可以打开输入关节目标位置/输入末端目标位置的窗口，输入目标位置数据后，按住“按住移动”按钮（或按住示教器左下角的“按住移动”按钮），机械臂会移动到目标位置，松开该按钮会停止移动。点击“取消”可关闭该窗口。



2. 坐标系

点击“坐标系”按钮可选择当前使用的坐标系以及手动移动的参考坐标系，默认会使用基坐标系，表示以机械臂底座为坐标基点。还可以选择工具坐标系或用户坐标系，工具坐标系表示以末端工具的末端点为坐标基点，用户坐标系由用户自由设定。

切换坐标系后，界面上的数据显示、移动末端时的移动方向以及编程时使用的坐标系都会相应改变。



注意

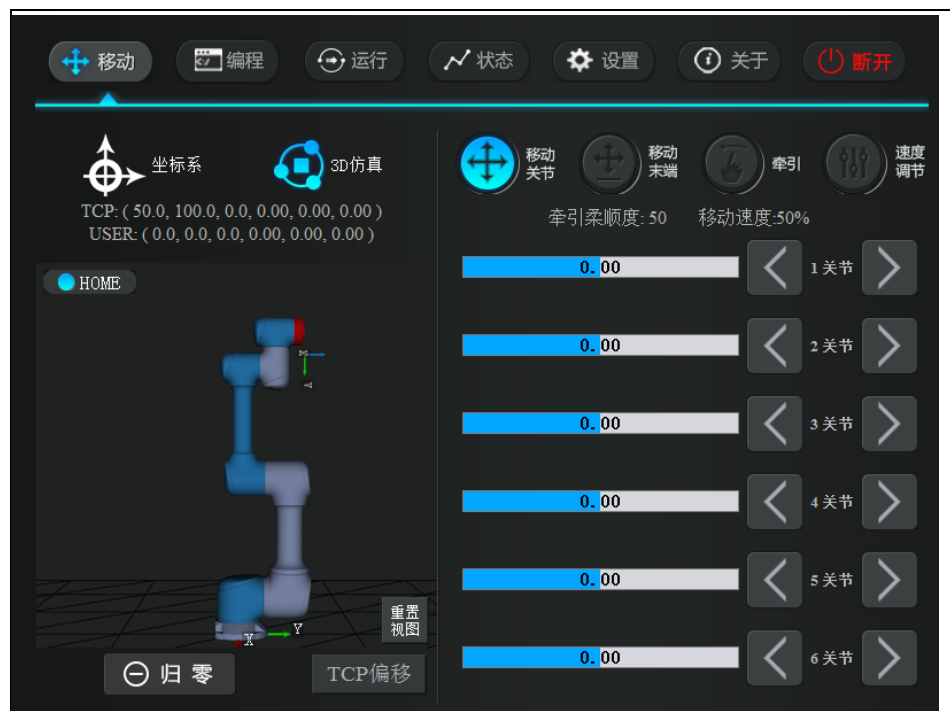
- 在界面中设置的坐标系，不在运行程序时生效。程序中需要使用设定坐标系的函数重新设定。

点击“工具坐标”、“用户坐标”后会跳到设置界面中设置工具坐标系的参数，详见设置界面。



3. 3D 仿真

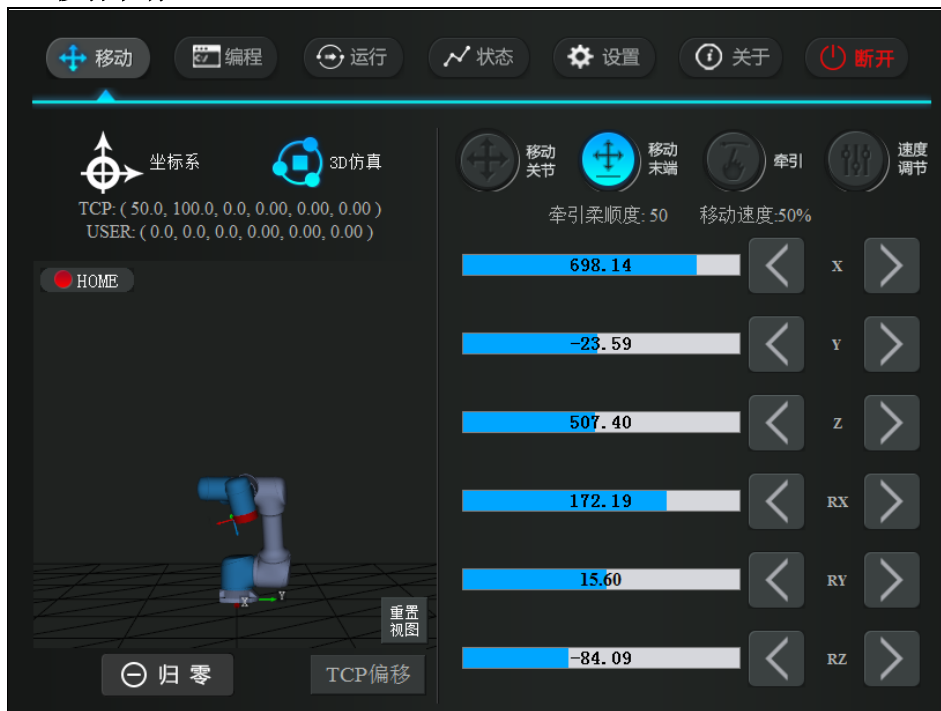
点击“3D 仿真”按钮会显示虚拟的仿真机械臂，仿真机械臂的姿态会和实际机械臂一样。仿真界面中会显示当前机械臂是否处于设定的 home 位置。



4. 移动关节

移动界面中默认会选择“移动关节”，蓝色进度条表示该关节当前位置，按住某个关节的左/右箭头按钮（或按住示教器右侧的“+”“-”钮，从上到下对应 1~6 关节）可移动该关节向正反两个方向旋转，松开按钮会停止移动。

5. 移动末端



点击“移动末端”按钮，切换到末端移动界面。按住某个方向的左/右箭头按钮（或按住示教器右侧的“+”“-”按钮，从上到下对应 X、Y、Z、RX、RY、RZ 六个移动方向），松开按钮会停止移动。

X、Y、Z 表示沿坐标轴方向的末端直线移动，RX、RY、RZ 表示沿坐标轴方向的末端旋转。



注意

- 移动末端运动时务必在坐标系选项下确认当前运动的参考。

6. 牵引

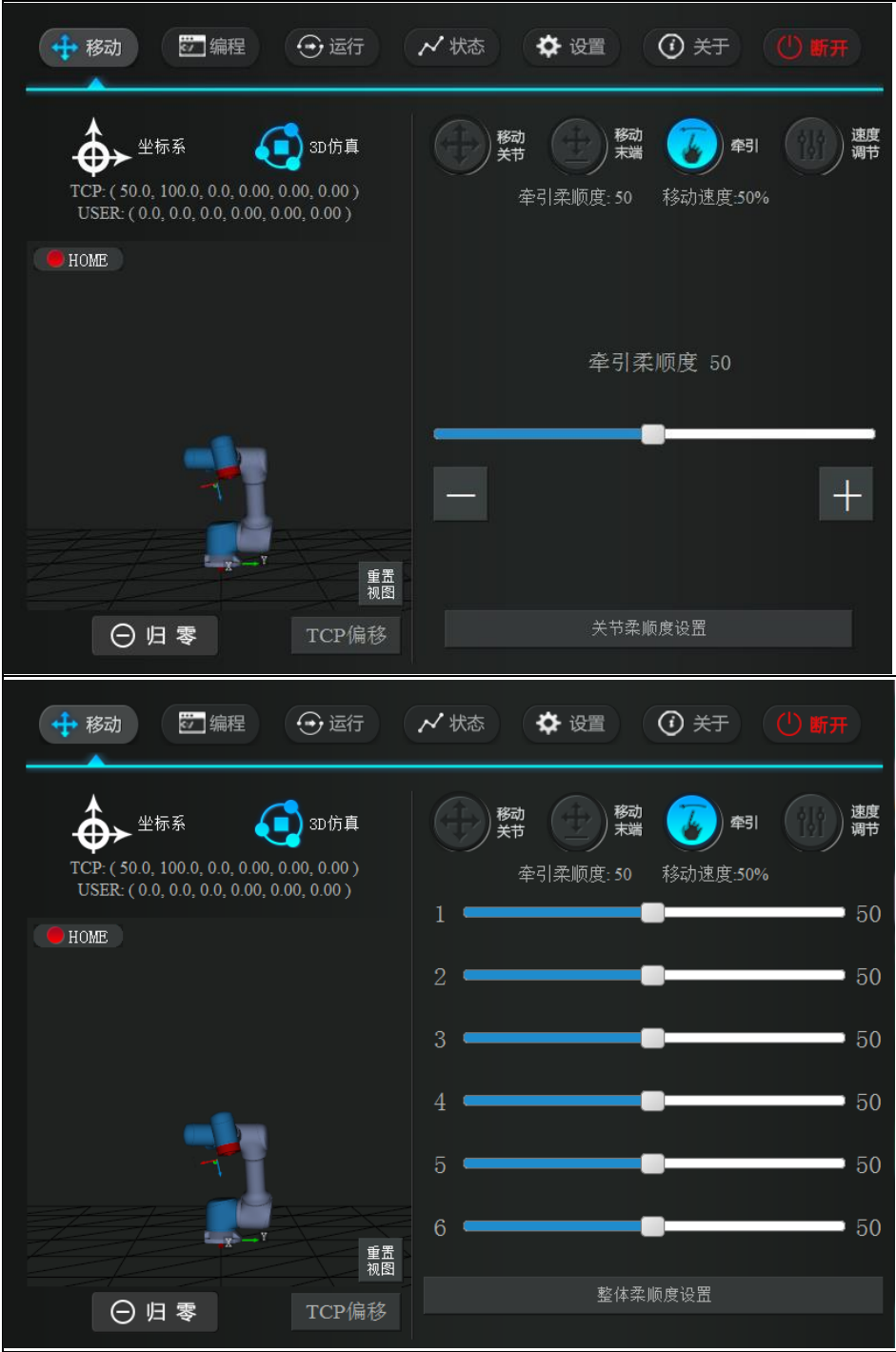
在机器人非运行状态下按住末端牵引示教的按钮，等待末端指示灯变成绿色闪烁状态，即可进入牵引示教模式。此时用户可以拖动机器人的任意关节进行移动。松开按钮后即退出牵引示教模式。

在牵引模式下，可以在“移动”-“牵引”界面上拖动柔顺度滑块调整整臂的牵引柔顺度，柔顺度越大，整臂牵引的柔顺性越好。此外，如果只需改变某个关节的柔顺度，可以在“关节柔顺度设置”界面下单独调整每一个关节的柔顺度。



注意

- 在牵引模式下，当关节接近限位时会产生反向驱动力以提示用户接近极限位置。但是用户如果施加足够大的力仍可以使关节继续接近极限位置。当关节超过极限位置时机器人会报错并停机。



7. 速度调节



点击“速度调节”按钮会出现速度条，拉动速度条可以改变移动关节、移动末端及归零速度。也可以按住示教器右侧最下面一组“+”“-”按钮，快速调节速度。

8. 归零按钮

按住界面左下角的“归零”按钮，机械臂会移动到设定的 Home 点，松开按钮会停止移动。默认零位是每个关节角度值都为 0，机械臂呈直立状态。

9. TCP 偏移

点击界面下方中部的“TCP 偏移”按钮，会显示当前末端在工具坐标系下的偏移量。点击“从此点开始计算 TCP 偏移量”，会将当前点设为 TCP 偏移量计算的原点。



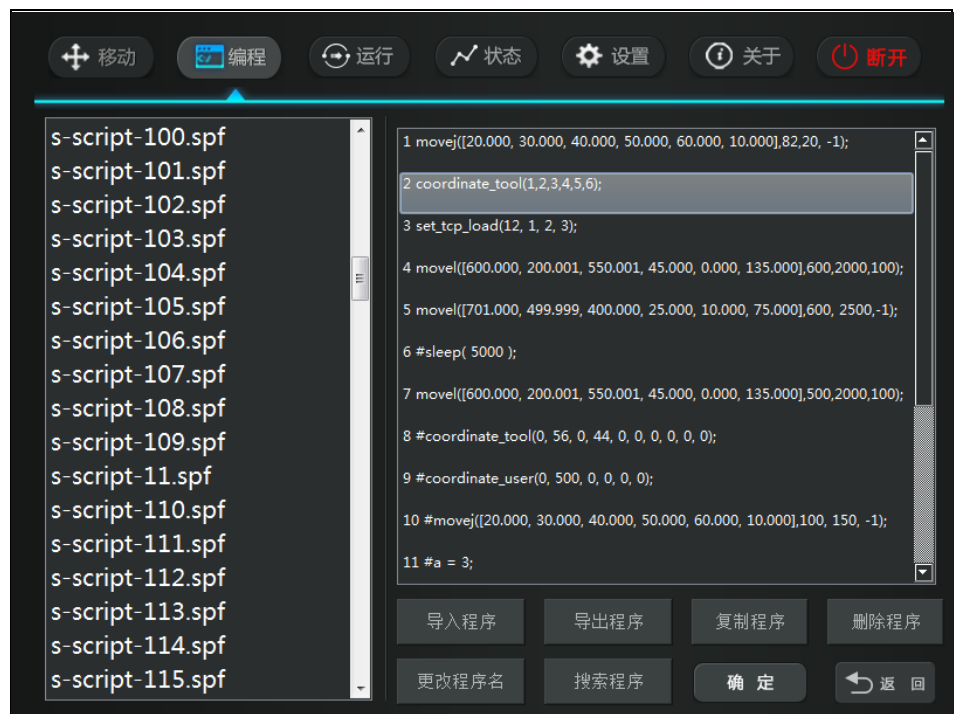
3.2 编程界面

点击上方导航栏中的“编程”会进入编程界面。

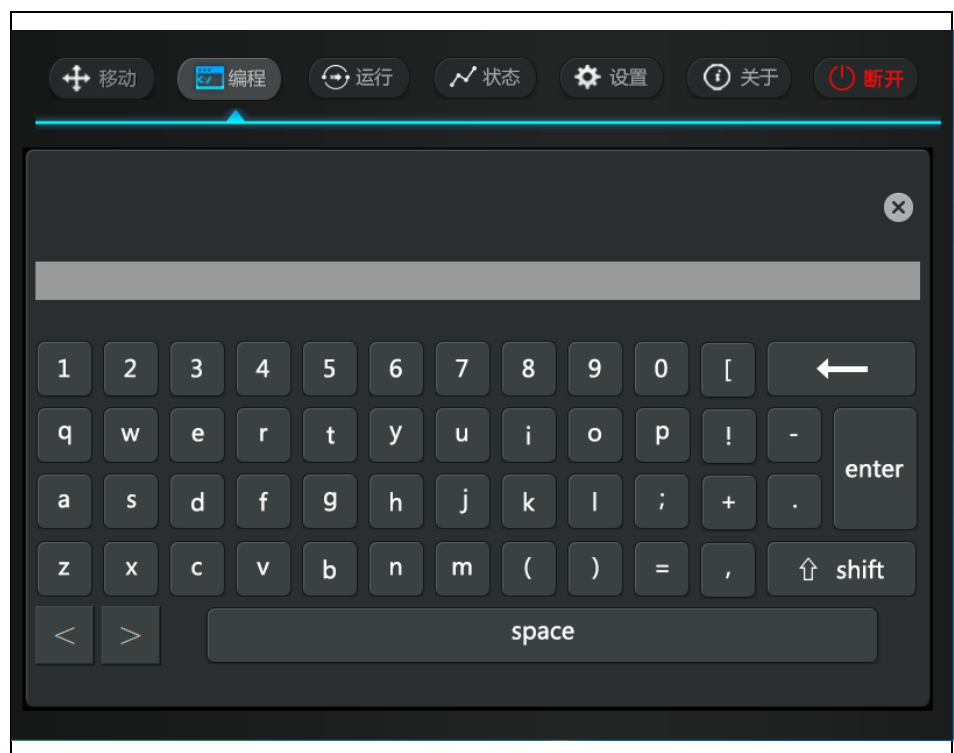


打开程序，选择已有程序点击“确定”打开，可以预览现有程序。也可点击“导入程序”按钮从外接设备中选择已经编写好的 spf 程序文件，导入系统文件库中。导入后界面左侧的程序列表中会多出刚刚导入的程序。点击“导

出程序”可以将选中程序导出到 U 盘。点击“复制程序”可创建一个选中程序的复制，添加到列表中。点击“删除程序”可删除选中的程序。



创建程序，输入程序名。



创建好程序后会进入到编辑程序界面，此处将程序命名为“siasun”。



新创建的程序会以“程序名.spf”格式保存，点击左上方的“放弃编辑”按钮可退出程序编辑，按钮右侧为当前打开的程序名。

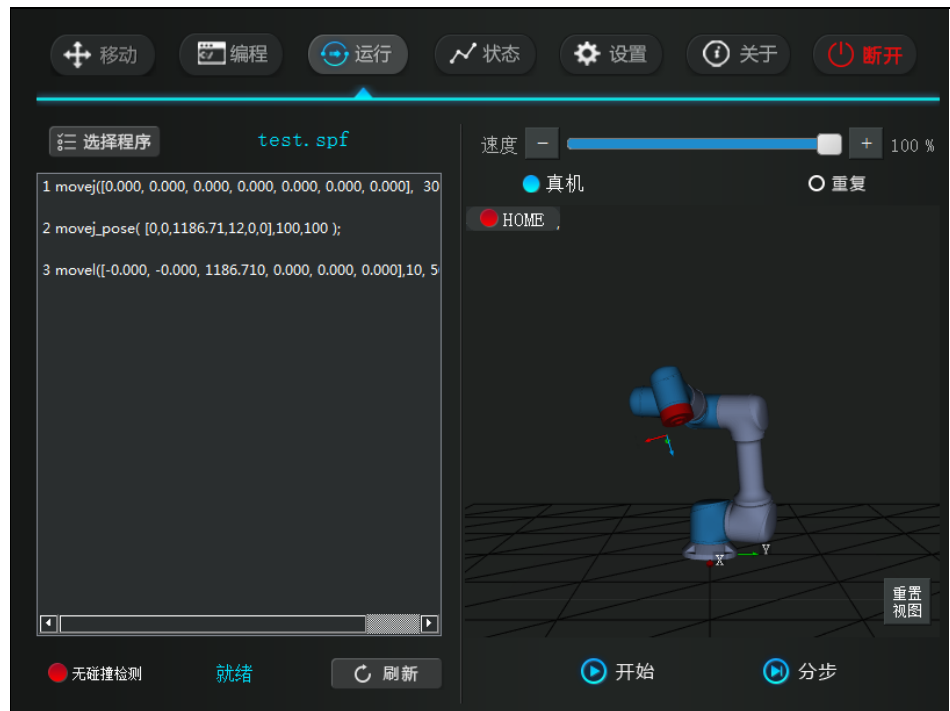


注意

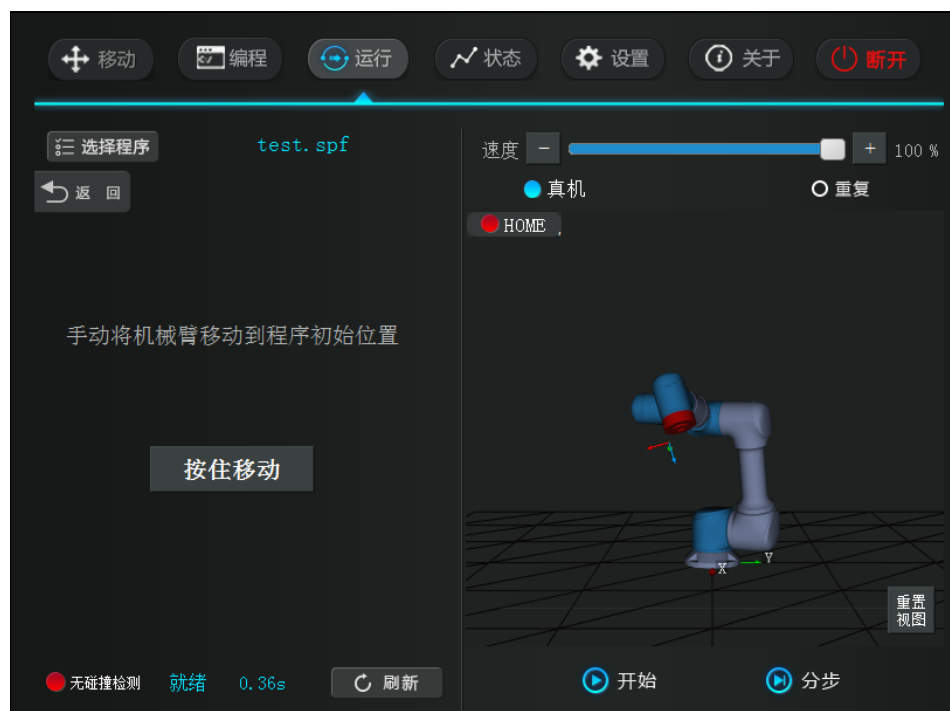
- 程序名只允许使用字符、数字、下划线。

3.3 运行界面

切换至运行界面，点击“选择程序”可以打开程序。



点击右下方的“开始”按钮来运行程序。程序运行时会判断机械臂当前位置和程序起始位置是否重合。为保证安全，若机械臂当前位置与程序起始位置不重合，会让用户手动将机械臂移动到起始位置，以下界面会跳出，用户按住“按住移动”按钮，机械臂会自动移动到程序起始位置。





注意

- 若通过远程的方式运行程序，机械臂不会判断当前位置是否在程序起始点，用户务必确认运行路径的安全性。

移动到起始位置后，再次点击“开始”按钮，机械臂便开始运行程序。

也可以点击“分步”分步执行程序，此时程序会逐条执行程序语句，每执行完成一条语句后暂停，等待用户再次点击“分步”执行下一条语句。

若选择了重复运行程序，或者由外部指令触发程序时，机械臂会自动从当前位置移动到程序起始点，用户务必评估该移动中的安全性。

运行时可暂停或停止。上方的速度条可改变运行速度百分比。点击“重复”可循环运行程序或设置重复次数。运行时末端灯带会变为绿色。

左下角显示了当前是否开启碰撞检测功能，红色点代表未开启，蓝色点代表开启。可在设置中切换状态。

点击“真机/仿真”按钮可以选择当前运行为在真机上运行还是在仿真环境下运行。



3.4 状态界面

点击上方导航栏中的“状态”会进入状态界面。

状态界面会显示当前末端的位置、各关节的角度以及一些系统信息。

可以查看 I/O 输入的状态及设置 I/O 输出的状态，红色表示低电平，蓝色表示高电平。

控制柜上有 8 个 input I/O，8 个 output I/O。机械臂末端有 2 个 input I/O，2 个 output I/O。点击“Output 输出”中的圆点便可切换高低电平状态。

在日志中可查看机械臂每天的运行日志



移动

编程

运行

状态

设置

关于

断开

关节及末端数据

日志

控制柜数字I/O

工具I/O

siasun_log_2017-06-19.txt
siasun_log_2017-06-16.txt
siasun_log_2017-06-14.txt
siasun_log_2017-06-13.txt
siasun_log_2017-06-12.txt
siasun_log_2017-06-09.txt
siasun_log_2017-06-08.txt
siasun_log_2017-06-07.txt

2017-06-19 17:32:44; read init parameters; axis bias: -34.6111 -305.314 -285.423 -344.208 -3

2017-06-19 17:32:44; read init parameters; DH: 0.3117 0.4 0.4 0.1753

2017-06-19 17:32:44; read init parameters; install: 90 0

2017-06-19 17:32:44; read init parameters; limits_velo: 130 125 185 185 185 185

2017-06-19 17:32:44; read init parameters; limits_acc: 1600 1600 2000 2000 2000 2000

2017-06-19 17:32:44; read init parameters; limits_Pos: 180 105 180 115 180 110 180

2017-06-19 17:53:04; read init parameters; axis bias: -34.6111 -305.314 -285.423 -344.208 -3

2017-06-19 17:53:04; read init parameters; DH: 0.3117 0.4 0.4 0.1753

2017-06-19 17:53:04; read init parameters; install: 90 0

2017-06-19 17:53:04; read init parameters; limits_velo: 130 125 185 185 185 185

2017-06-19 17:53:04; read init parameters; limits_acc: 1600 1600 2000 2000 2000 2000

17



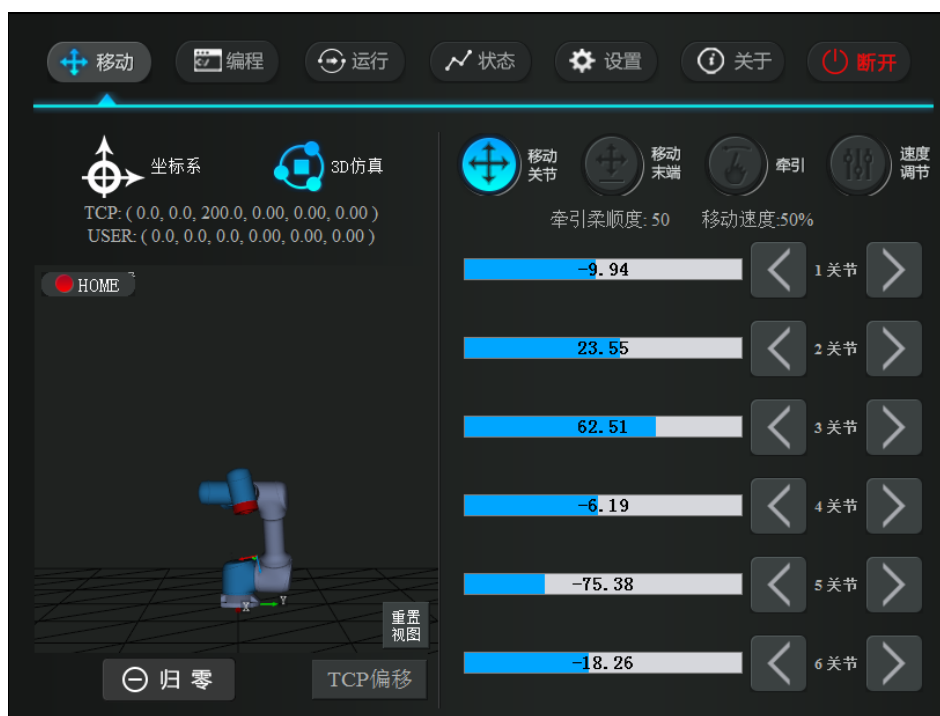
3.5 设置界面

点击上方导航栏中的“设置”会进入设置界面。

工具及 TCP 设置:



工具坐标系设置中可设置工具坐标相对于法兰面的偏移量，同时可以设置工具质心位置和质量，可以保存多个工具坐标系的设置。设定工具坐标系后，仿真模型中会显示出对应的工具坐标系：



用户可以使用负载质心辨识功能对负载质心进行辨识。机器人会进行一系列的运动，根据运动过程中记录的数据可以对负载的质心进行计算：

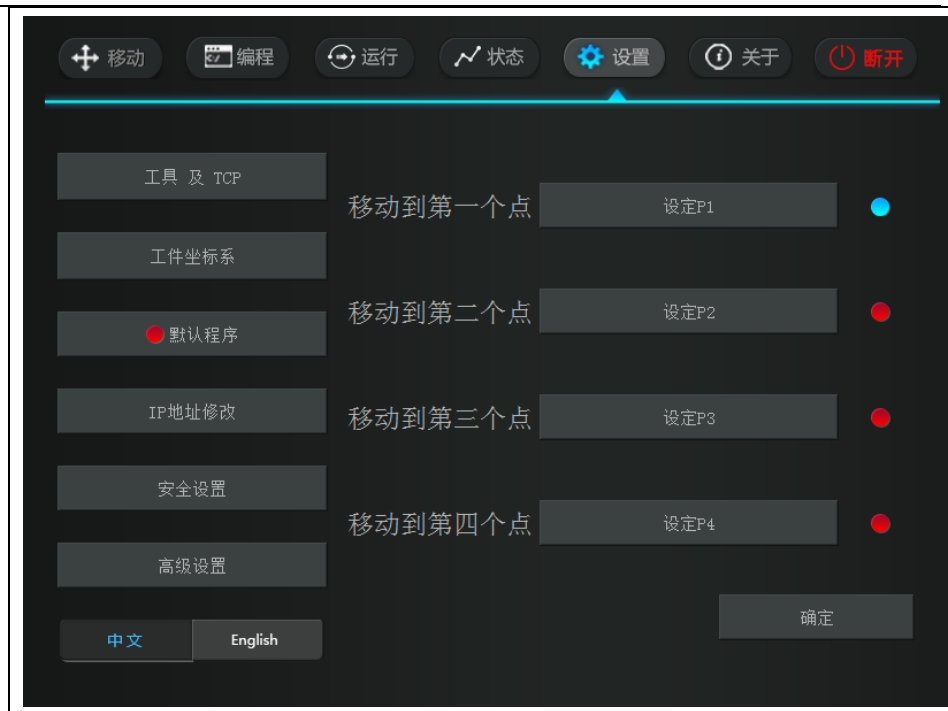
负载辨识功能运行时，机器人的 4、5、6 关节会自动调整至辨识姿态。随后 6 关节在 -90° 至 $+90^{\circ}$ 之间运动。6 关节运动停止后回到 0° 位置，随后 5 关节在 -90° 至 $+90^{\circ}$ 之间运动。5 关节运动停止后，机器人的 4、 5、 6 关节自动调整至负载辨识功能运行前的位置，辨识结束。用户务必评估运动路径的安全性。

点击“读取辨识位置”可以在界面上显示辨识得到的负载质心位置。

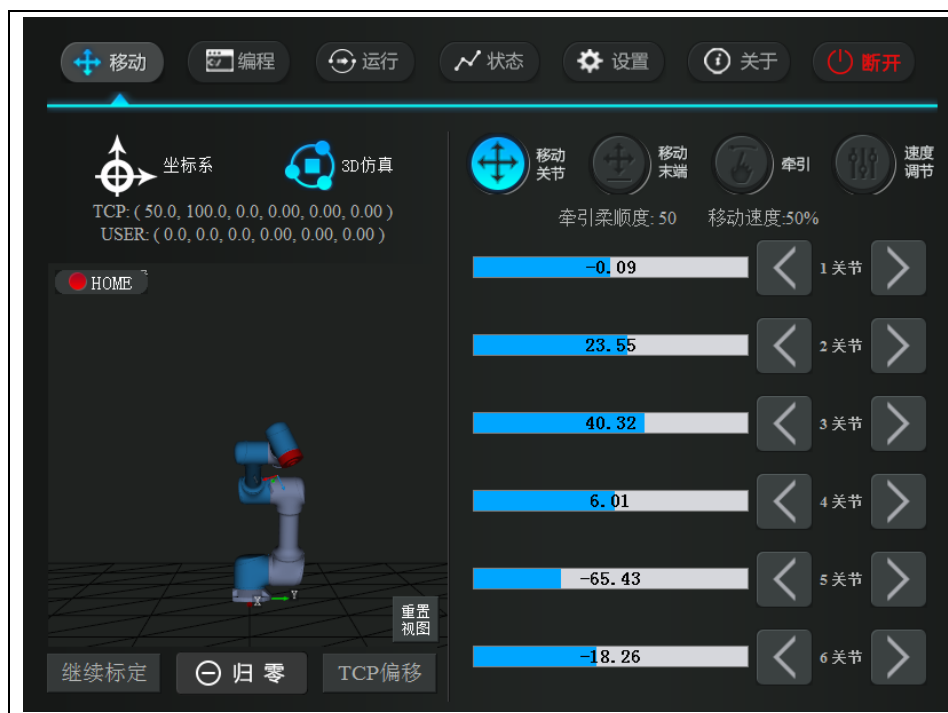
点击“TCP 标定”可进入末端工具位置标定功能界面：



可选择 4 点标定法或 6 点标定法，按照提示进行工具末端点标定。



点击设定某个标定点后，会跳转到移动页面。移动机械臂到目标点后，点击左下角的“继续标定”按钮，继续标定下一个点。



注意

- 利用 4 点法、6 点法进行 TCP 标定时，务必确认当前 TCP 设置为 0.

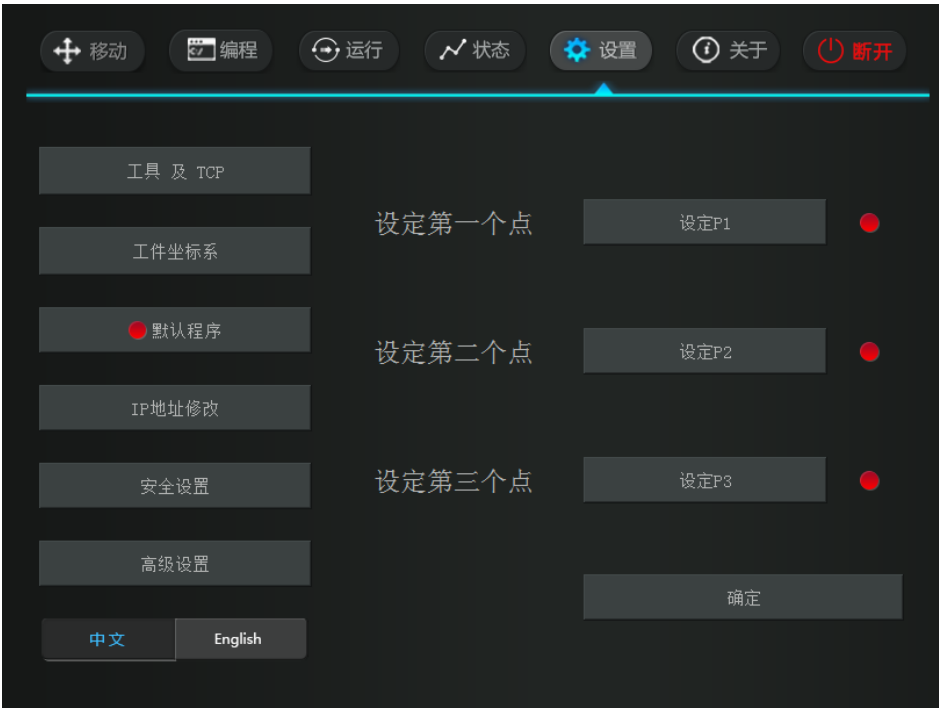
工件坐标系设置：



工件坐标系设置中可设置工具坐标相对于基坐标系的偏移量，可以保存多个工具坐标系的设置。

点击“工件坐标系标定”使用三点法标定工件坐标系，P1 点为工件坐标系的原点，P2 点指定工件坐标系的 X 方向，P3 点指定工件坐标系的 Y 方向。

设定工件坐标系后，仿真模型中会显示出对应的工件坐标系。

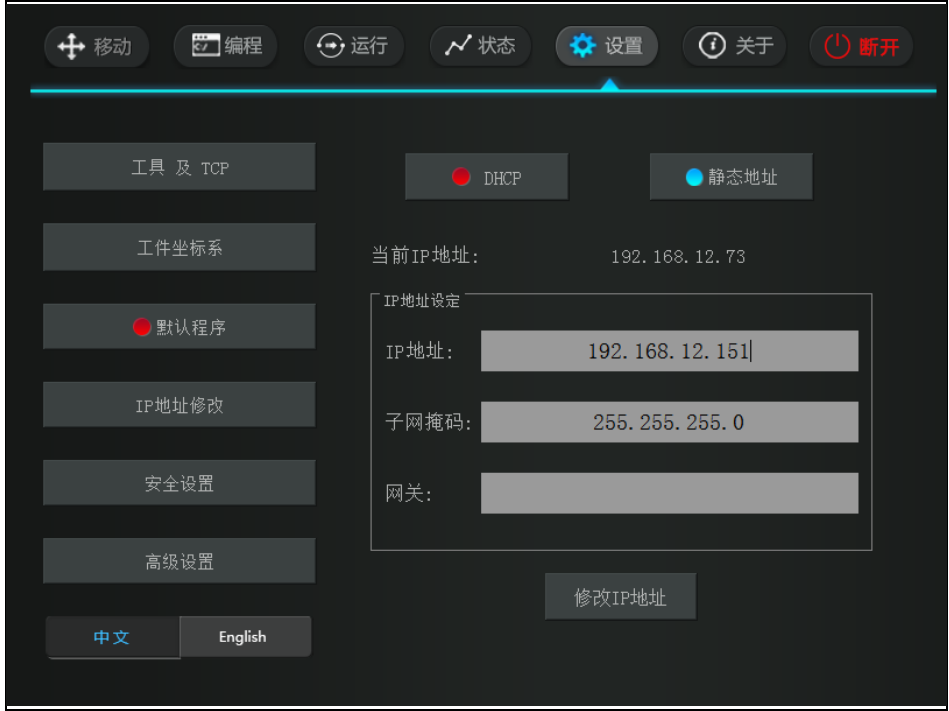


默认程序:



默认程序功能可选定开机时自动加载的默认程序，可使用系统 IO 或 modbus 触发该程序运行。若设置了默认程序，“默认程序”按键左侧的圆点会变为蓝色，若系统未设定默认程序，圆点会变为红色。

IP 地址修改:



可显示本机的当前 IP 地址，设置使用静态或 DHCP 自动分配 IP 地址。

安全设置：

输入密码后可开启。可设定碰撞检测功能是否开启，调整灵敏度以及设定虚拟边界。



安全校验码是由当前安全配置信息计算出的安全码，当配置信息有任何改动，安全校验码都会改变。机器人系统启动时会检测当前的安全配置信息是否与安全校验码匹配，若不匹配，系统会发出报警，用户需确认安全配置信息。

碰撞检测：

用户可以勾选“碰撞检测”以启用碰撞检测功能，此时状态指示按钮为蓝色。当碰撞检测功能被启用时，在机器人运行过程中如果发生碰撞，机器人会立即停机直到用户确认安全且错误信息被清除。



- 如果在机器人的运动过程中出现碰撞检测误报的现象，请先停止机器人运动，并降低碰撞检测灵敏度至误报现象消除。



虚拟边界:



系统共可设置 6 个虚拟边界，包括 2 个水平边界，4 个垂直边界。水平边界需要输入边界平面与基坐标 XY 平面的距离（单位 mm）。垂直边界需要输入边界平面与基坐标 Z 轴的距离（单位 mm），以及绕 Z 轴旋转的角度（单位°）。垂直平面基准为基坐标系 YZ 平面。

例如设定了一个与 XY 平面距离 1100mm 的水平边界，以及一个与 Z 轴距离 300mm，旋转 45° 的垂直平面。在三维仿真模型中会显示出虚拟边界与机械臂之间的位置。



当机械臂触碰到虚拟边界时，系统会发出警告，若正在运行程序，程序会被暂停。

当设定了工具坐标系后，工具末端点会显示出工具坐标系，当工具末端点触碰到虚拟边界时，系统会发出警告，若正在运行程序，程序会被暂停。

安全 IO 配置：

安全 IO 配置部分可选择安全输入的信号种类及安全输出的信号意义。

可配置安全输入包含保护性停止（1 类）、保护性停止（2 类）、安全减速。

保护类型	描述	复位方式
保护性停止(1 类)	受控的停机，停机后抱闸松开，不保证机械臂停在原有轨迹上	手动
保护性停止(2 类)	受控的停机，抱闸保持，机械臂停在原有轨迹上	手动/自动
安全减速	速度降低至安全运行速度范围	手动/自动



高级设置:

输入密码后可开启高级设置。

安装设置用于设定机械臂安装方向。

自动登录打开后，机械臂会在通电后自动上电、松抱闸，进入运行待命状态。

恢复出厂设置会将所有软件配置信息恢复到初始状态。

三位置使能开关可在此选择是否使用。

“设置当前点为归零点”可以将当前点设置为归零时移动到的点。

修改密码功能可更改软件当前的密码，初始密码为 111。

软件更新功能可更新机械臂软件。



中、英文语言切换：
切换面板中、英文显示。

3.6 关于界面

点击上方导航栏中的“关于”会进入关于界面。

关于界面提供关于软件版本的信息。



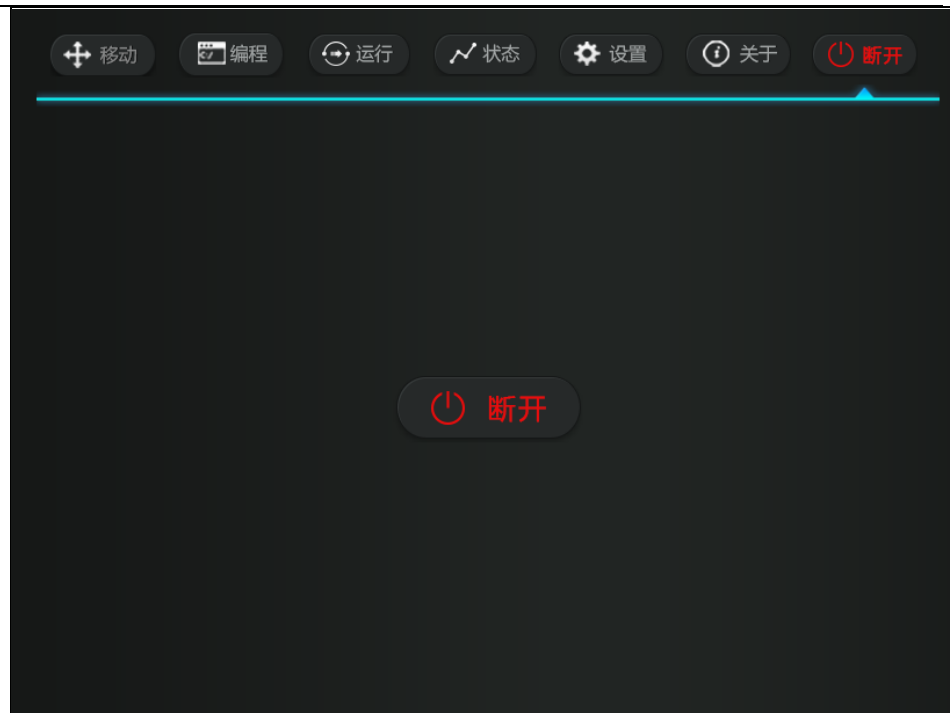
3.7 断开界面

需要断开与机器人的连接时，点击屏幕右上角的“断开”按钮，再点击屏幕中间的断开按钮，抱闸便会卡住。程序会回到欢迎页面。在欢迎界面点击“开始”按钮可重新连接机械臂。



注意

- 点击断开按钮，抱闸卡住时，机械臂有可能会下落一小段距离。

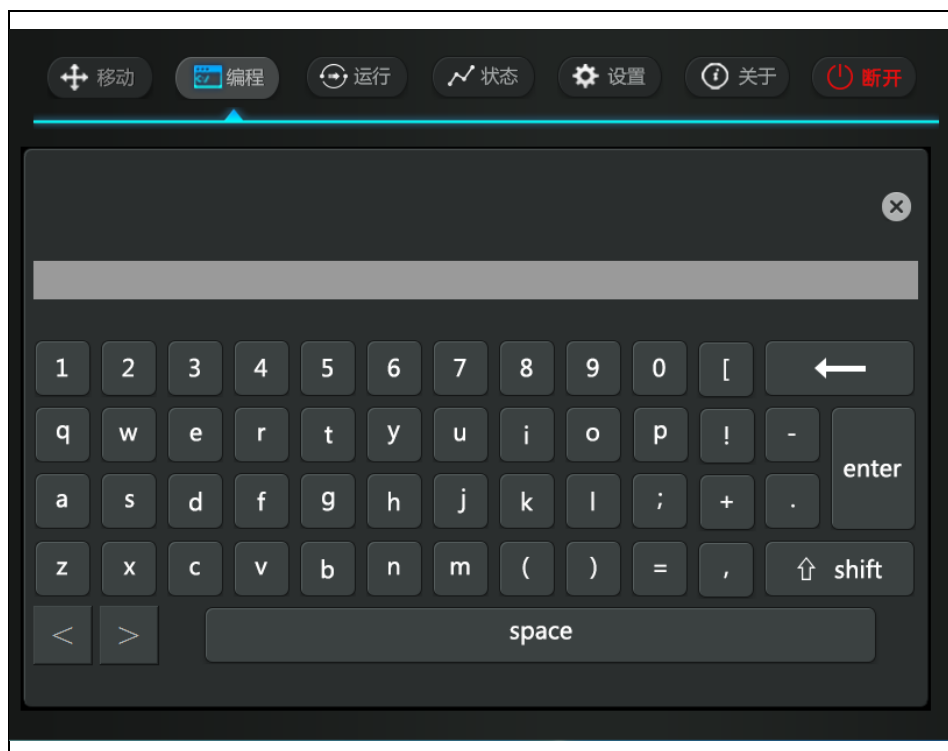


4 机器人脚本编程

4.1 创建程序



创建程序，输入程序名。

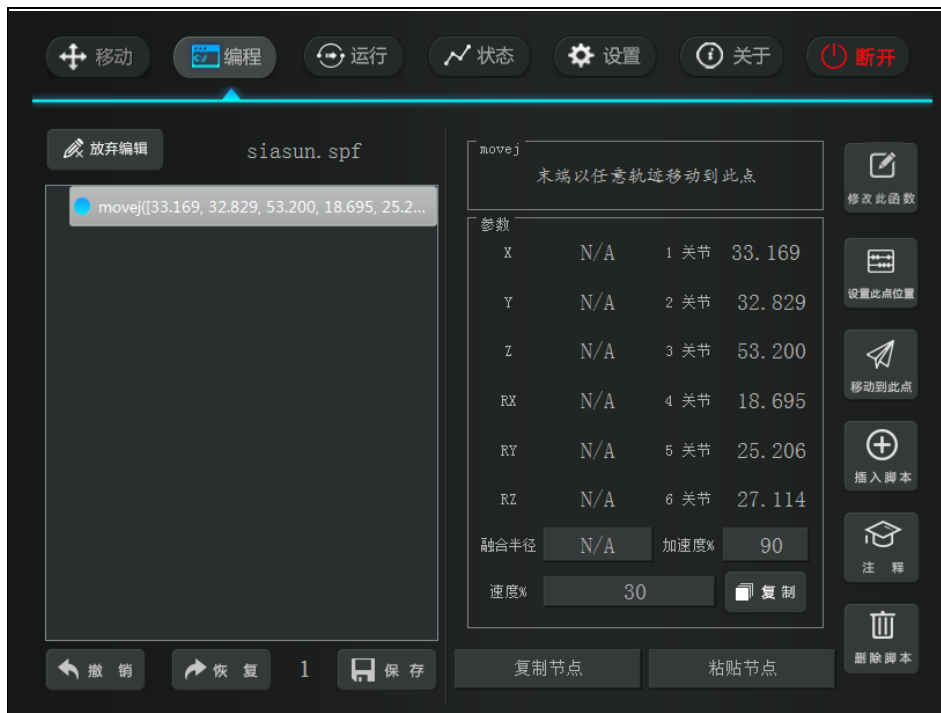


创建好程序后，选择脚本语言编程，会进入到编辑程序界面，此处将程序命名为“siasun”。



注意

- 程序名只允许使用字符、数字、下划线。



新创建的程序会以“程序名.spf”格式保存，点击左上方的“放弃编辑”按钮可退出程序编辑，按钮右侧为当前打开的程序名。

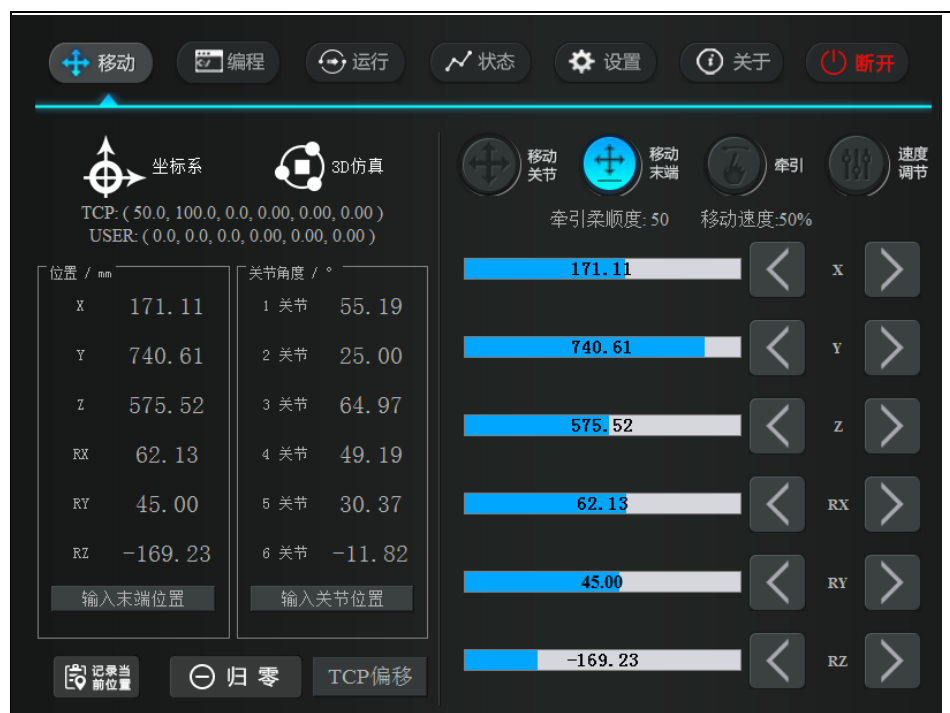
左侧为程序显示列表，当创建了一个新程序后，会自动添加“movej”节点，并保存当前机械臂的位置到该点上。在程序中，每一个节点表示机械臂的一个位置。程序运行一个节点就是移动到该节点的位置。运行程序时机械臂会从运行程序前所在的任意位置移动关节到程序的第一个点。

右侧有“movej”点的功能描述及位置信息，包含了末端空间位置、关节位置、速度。点击速度可以更改运动速度，编程时默认为 30%。movej 的速度为关节最大角速度百分比，范围：1-100%，加速度为关节最大角加速度百分比，范围：1-100%。movei、movec 的速度为末端速度，范围：1-1000mm/s，加速度单位：mm/s²。

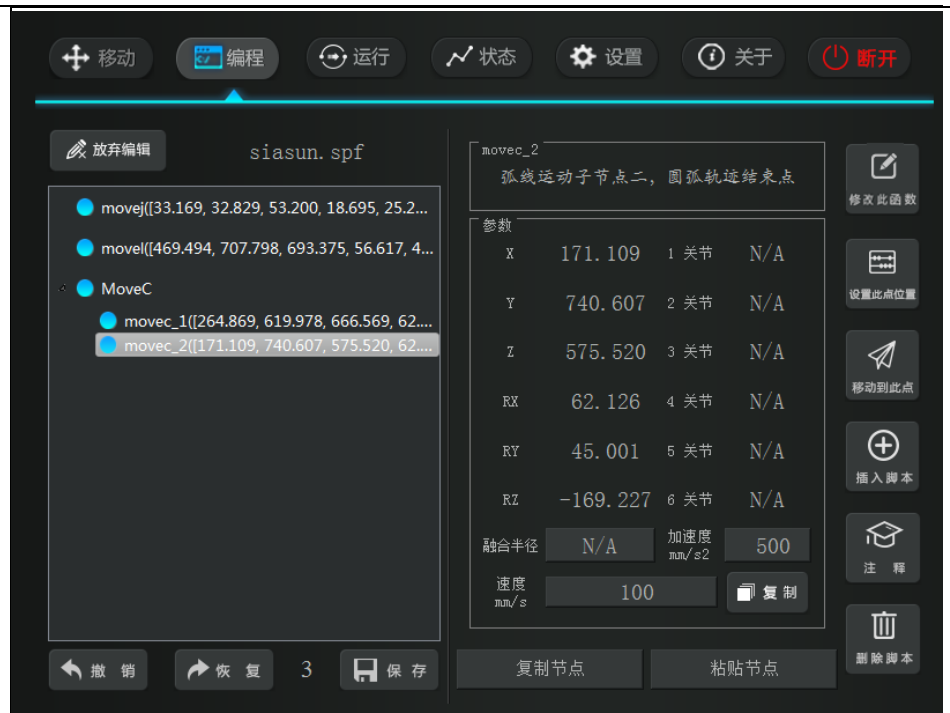


点击右侧的“修改此函数”，可以手动修改脚本指令。

点击右侧的“设置此点位置”或双击想要设置位置的节点便可设置/更改节点的位置，此时页面会跳转到移动界面（左下方会出现“记录当前位置”按钮）：



将机械臂移动到目标位置后，点击左下角的“记录当前位置”按钮，程序会跳转回到编程界面，节点的位置已被设置/更新。



点击“移动到此点”，机械臂按照关节的形式运动到目标点。

点击“插入点”按钮，会跳出添加节点窗口。

添加节点窗口的上方导航栏中有 3 个标签，点击标签可切换。

4.2 move 标签：移动指令集合



- **movej**: 机械臂移动关节到各关节的目标点, 不考虑末端轨迹 (MoveJ 与其功能相同, 但所需输入参数更多), 点击添加该节点到程序中。
- **movel**: 机械臂末端沿直线移动到此点, 点击添加该节点到程序中。
- **movej_pose**: 给定一个目标点位置, 机械臂通过匀速移动各关节的方式, 移动到目标点。此函数输入参数与 **movel** 相同, 但移动方式与 **movej** 相同。
- **movec**: 机械臂末端弧线运动, 点击后在程序中添加的节点会有 2 个子节点, 第一个子节点为圆弧轨迹的中间点, 第二个子节点为圆弧轨迹的结束点。
- **move_phi**: 机械臂保持末端位姿不变, 臂形角移动一个给定的增量。
- **tcp_move**: 机械臂末端按照工具坐标系平移给定的偏移量。
- **move_camera**: 用于使用相机拍照抓取。通过示教 P1 点 (拍照点), P2 点 (抓取点), 填写 **offset** (相机计算出的偏移)。来实现拍照定位抓取功能。
- **牵引复现**: 跳转到牵引复现界面, 选择要存放轨迹的序号, 点击“确定”后机械臂进入准备状态, 此时按下机械臂末端牵引示教按钮, 可以开始拖拽机械臂并记录轨迹, 松开末端牵引示教按钮, 系统停止记录轨迹。轨迹存放点中蓝点表示未被占用, 红点表示已被占用, 重新示教将覆盖原轨迹, 程序中将复现记录的轨迹。



- “脚本语言”按钮可添加脚本函数, 点击后会弹出软键盘, 用来编辑函数。关于脚本函数的介绍请参考《新松机器人脚本函数手册》。

4.3 functions 标签：函数集合

movefunctionssocket创建变量脚本语言

sleepIO

控制语句坐标系 工具

print添加注释

获得数据modbus

IO 功能函数:

movefunctionssocket创建变量脚本语言

io_outio_in

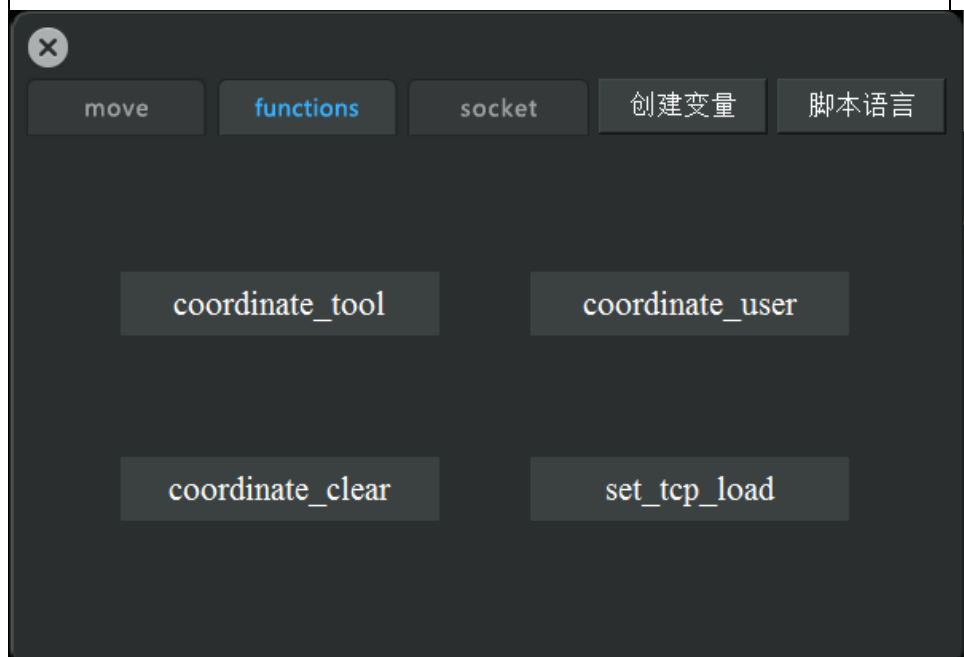
tool_io_outtool_io_in

io_v_outio_v_in

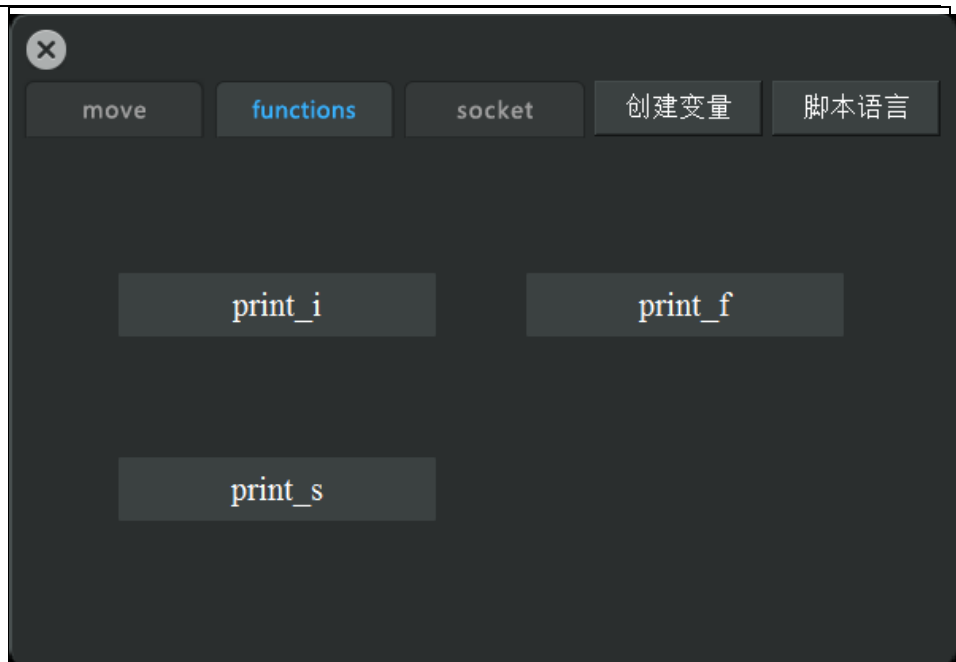
控制语句:



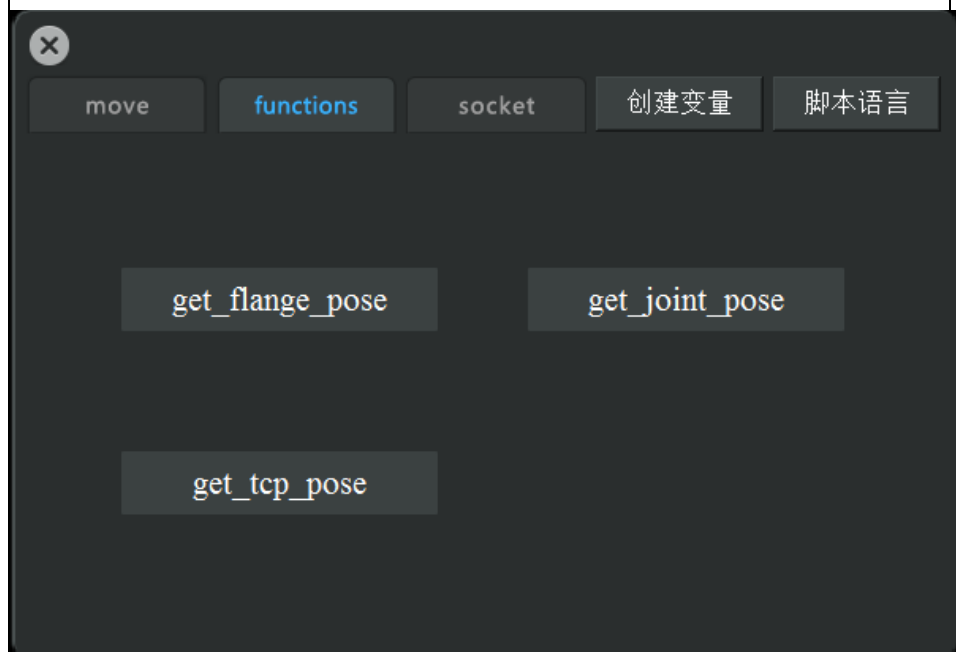
坐标系/工具:



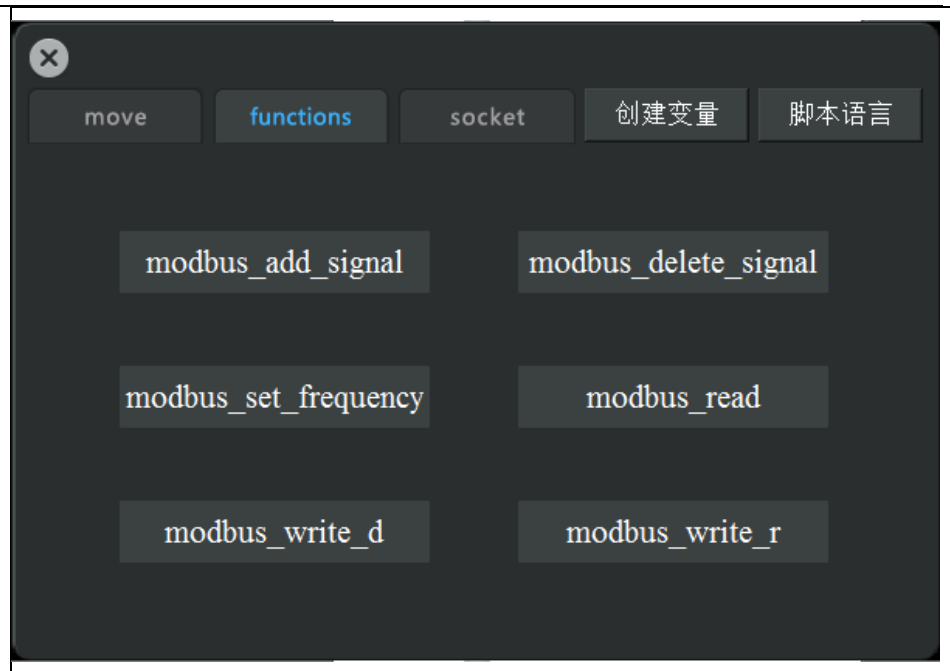
打印函数:



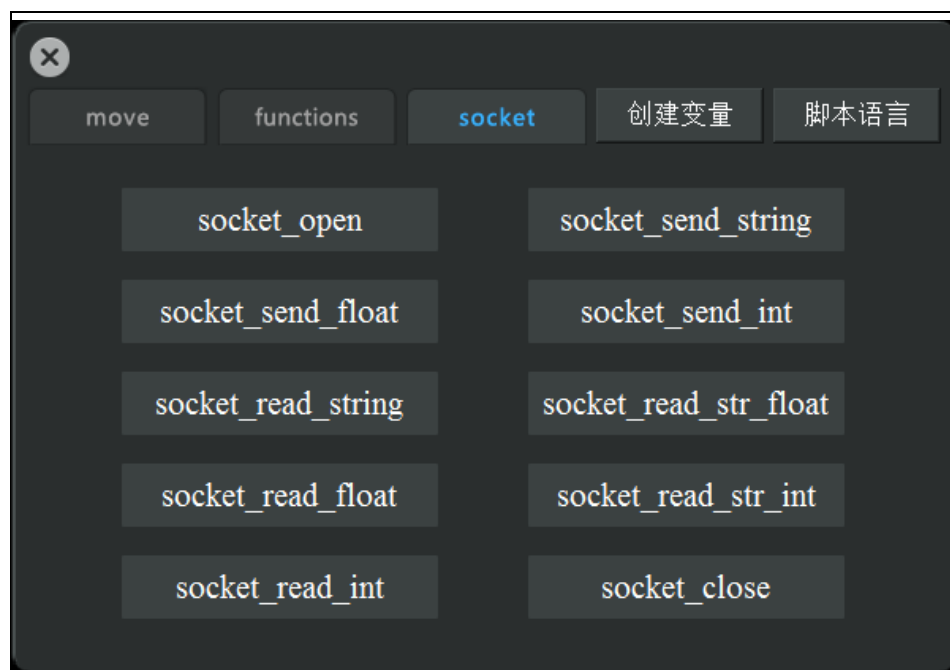
获得数据的函数：



Modbus 函数：



4.4 socket 标签：网络通信指令集合



- 以上函数为快捷输入，全部函数内容及相关功能详细解释请参考脚本语言说明中的网络通信部分。

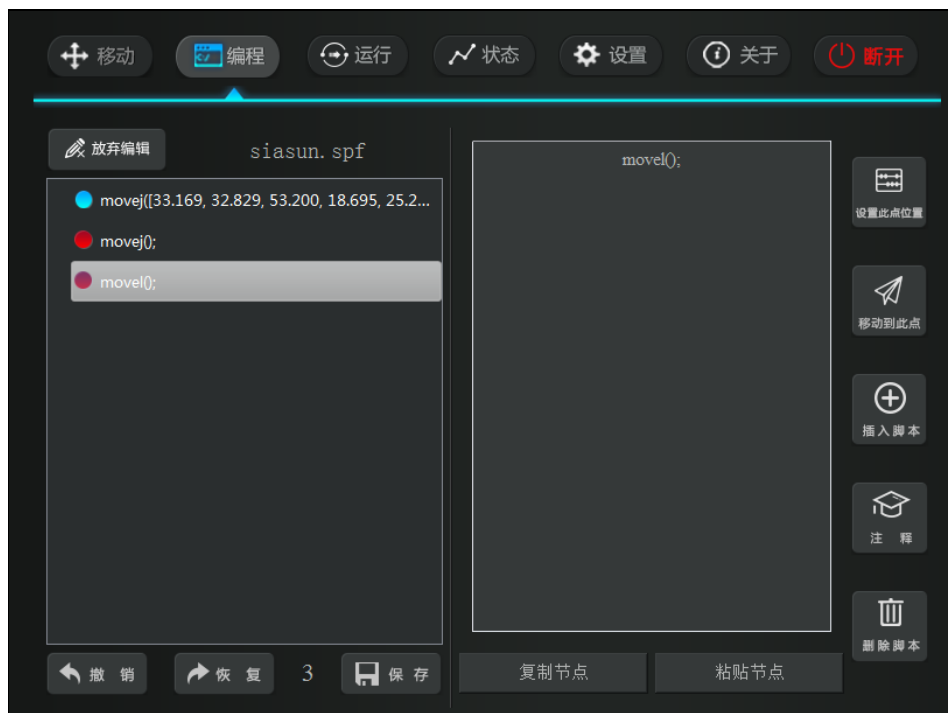
4.5 创建变量标签：

可以创建空间位置变量和关节列表变量，并且可以通过示教的方式记录点位。

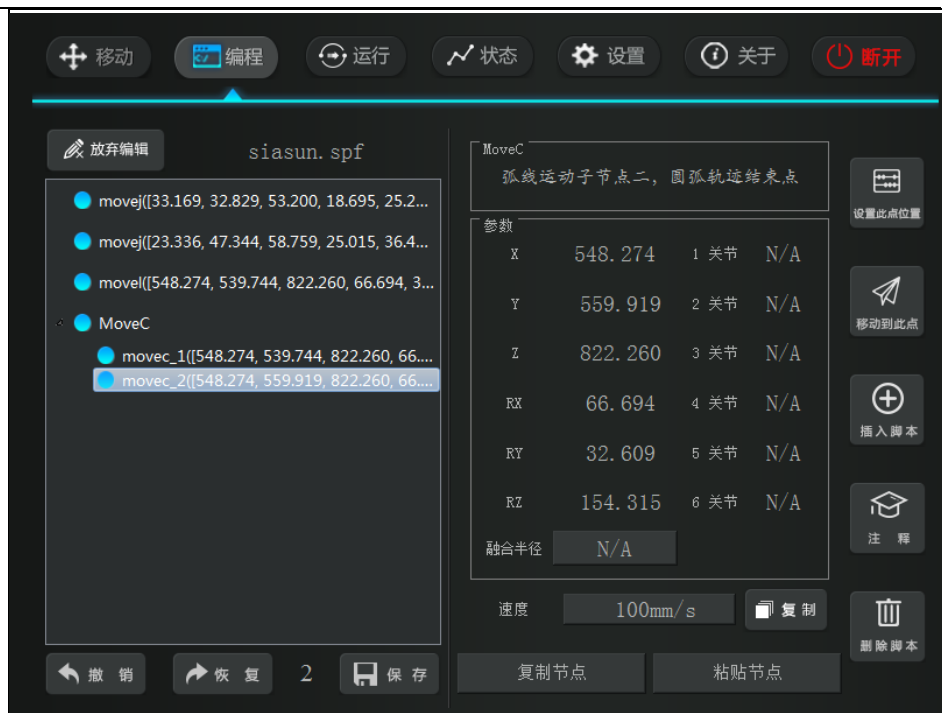


4.6 示例

各添加一个 movej，moveI 节点



设置两点位置，再添加一个 MoveC 节点并设置位置：



坐标系说明

若在脚本中未设定过坐标系，则末端运动函数的参数均表示法兰面中心点相对于基坐标系的位置。

若在脚本中使用 `coordinate_tool` 函数设定了坐标系，则该函数往后的所有末端运动函数的参数均表示工具坐标原点相对于基坐标的位置。

若在脚本中使用 `coordinate_user` 设定了工件坐标系，则后续函数中，原本相对于基坐标系的位置会变为相对于该工件坐标系的位置。

使用 `coordinate_clear` 函数可清除所有已设定的工具坐标系和工件坐标系。

按键功能

在左侧选择某个节点后，按住右侧的“移动到此点”按钮（或按住示教器左下角的“按住移动”按钮），机械臂会移动到此节点的位置，松开会停止移动。

“修改此函数”按钮可修改函数脚本。

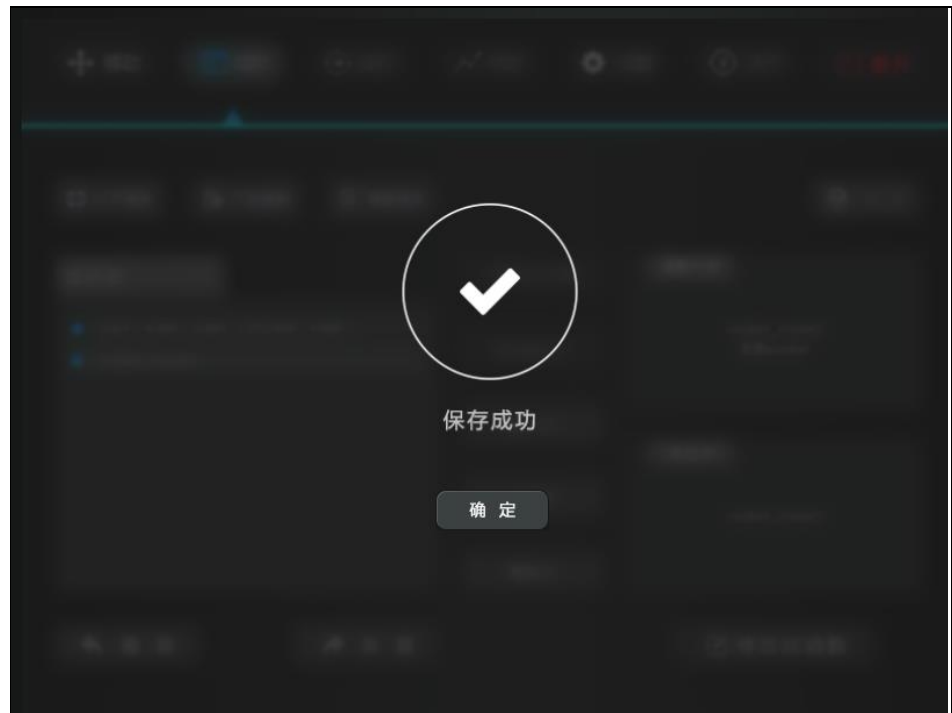
“注释”按钮可将某行节点注释掉，程序运行时跳过该节点，选择已注释的节点，点击注释按钮可取消注释。

点击“复制”按钮会复制左侧的速度，点击后再单击左侧想要修改速度的节点可快速批量修改速度。再次点击“复制”按钮会取消复制。

点击“复制节点”可复制当前选中节点，点击“粘贴节点”可将复制的节点粘贴到当前选中行的下一行。

点击“结尾添加过渡”或者“融合半径”可以给任意运动之间添加融合半径，单位为 mm。

点击“保存”按钮来保存程序。



5 机械臂外部通讯接口

TCP/IP:

机械臂在上电状态下会作为 TCP Server 监听端口：2000 和 2001 的连接。端口 2000 可接收对机械臂的控制命令及返回状态。端口 2001 会以 10Hz 的频率向外部发送机械臂当前的状态信息。

2000 端口:

2000 端口采用请求-响应式的应答机制,可以接受脚本指令及一些特殊指令, Client 和 Server 间的通讯协议如下:

Client 向 Server 发送机械臂脚本命令:可发送一句或一段脚本让机械臂执行。

发送一段脚本指令时,由<start>标识符开通,以<end>标识符结尾。机械臂会运行在<start>和<end>标识符之间的指令。整个脚本可以分多次发送完毕。

接收到<start>标识符后,会返回 Prog mode start 回执;接收到<end>标识符后,会返回 Prog mode end 回执;若脚本指令大小超过单次最大限制(8192 个字节),会返回 Exceed max size 8192 回执,并忽略已收到的<start>标识符。

Client 向 server 发送: run(程序名.spf) 或 run(程序名.spf , 速度百分比)。机械臂会执行相应程序,不设置速度百分比或者速度百分比参数无效时,使用当前速度运行。

如果当前机械臂处于空闲状态,会执行相应命令并在开始执行时向 client 发送 Script start 回执。如果当前机械臂处于非空闲状态,会向 client 发送 System not ready 回执。若机械臂处于锁定状态,会反馈 System lock 回执;若程序不存在,会返回 Program doesn't exist 回执。

机械臂在执行接收的脚本指令时,若检测到不符合语法规则,会返回 Syntax error 回执,不需要通过 reset 复位;若出现系统错误,返回 System error 回执,机械臂进入错误状态,需要通过 reset 指令复位;如果运行过程中触发碰撞检测,返回 Collision detect 回执,机械臂进入暂停状态,可通过 resume 指令继续程序。

机械臂在程序执行完毕时会向 client 发送: Script finish 回执。

Client 向 server 发送: speed(速度百分比),可修改程序运行的速度百分比。参数范围在(0,100],若修改成功,会返回 Speed set,若不成功,会返回 Speed invalid。

Client 向 server 发送: stop,机械臂会停止当前执行的指令。停止后会发送 Script stop 回执。若机械臂处于非运行状态,接收到 stop 命令后会发送 Already

stopped 回执。

Client 向 server 发送: pause, 机械臂会暂停当前执行的指令。停止时会发送 Script pause 回执。若机械臂处于非运行状态, 接收到 pause 命令后会发送 Already stopped 回执。

Client 向 server 发送: resume, 机械臂在暂停时会继续运行程序。运行时发送 Script resume 回执。若机械臂不在暂停状态, 接收到 resume 命令后会发送 Resume error 回执。

Client 向 server 发送: reset, 若机器人处于错误状态, 机器人系统会自动复位, 返回 Reset success 回执。若机器人未处于错误状态, 返回 No error to reset 回执。

Client 向 server 发送: login, 机械臂若处于断开连接状态, 返回 Login start 回执, 机械臂自动上电, 松开抱闸, 进入待命状态。登录成功后返回 Login success 回执, 登陆失败则返回 Login failed。若机械臂已处于登录状态, 则会返回 Already login 回执。

Client 向 server 发送: disconnect, 机械臂若处于连接状态, 返回 Disconnect success 回执, 机械臂断开连接。若机械臂已处于断开状态, 则会返回 Already disconnect 回执。若此时正在运行程序, 会返回 Stop program first 回执。

Client 向 server 发送: shutdown, 机械臂已处于断开状态, 机器人系统会自动关闭, 返回 Shutdown success 回执。若机械臂若处于连接状态, 返回 Must disconnect first 回执。

Client 向 server 发送: poweroff, 机械臂已处于断开状态, 机器人系统会自动断电, 返回 Poweroff success 回执。若机械臂若处于连接状态, 返回 Must disconnect first 回执。

Client 向 server 发送: status, 机械臂会反馈当前的 6 个关节位置及末端相对于基坐标系的位置。格式为: pos1 ; pos2 ; pos3 ; pos4 ; pos5 ; pos6; axis1 ; axis2 ; axis3 ; axis4 ; axis5 ; axis6 。

Client 向 server 发送: ikine(x, y, z, rx, ry, rz), 机械臂会按照当前的臂形角, 计算目标空间位置对应的各关节位置, 返回 Calculation done 回执, 并通过 2001 端口中的 IKineJointPose 变量输出 (单位: 度)。x y z 单位: mm, rx ry rz 单位: 度。计算成功后还会以字符串的形式返回计算后的逆解值, 格式为: axis1 ; axis2 ; axis3 ; axis4 ; axis5 ; axis6 , 缺少参数会返回 Parameter error 回执, 计算无有效解会返回 Calculation error 回执。例: ikine(100, 150, 200, 0, 45, 90)。

Client 向 server 发送: fkine(j1, j2, j3, j4, j5, j6,)，机械臂会计算出对应此关节的空间位置，返回 **Calculation done** 回执，并通过 2001 端口中的 FKineEndPose 变量输出（单位：mm）。参数单位：度。计算成功会后还会以字符串的形式返回计算后的末端位置，格式为： pos1 ; pos2 ; pos3 ; pos4 ; pos5 ; pos6 ，参数错误会返回 **Parameter error** 回执。例： fkine(20, 30, 40, 0, 45, 90)

若机械臂处于错误状态下接收到了外部指令，会返回 **System error** 回执。

若机械臂处于未就绪状态下接收到了外部指令，会返回 **System not ready** 回执。



注意

- 2000 端口所有返回的回执均以换行符\n结尾，用来判断是否有粘包发生。
- status\fkine\ikine 指令后续版本有可能删除。

2001 端口：

端口 2001 会以 10Hz 的频率向外部发送机械臂当前的状态信息。

数据列表如下：

意义	类型	变量数	字节大小	地址	注解
关节实际位置	double	6	48	0 - 47	6 关节各自实际位置
预留	double	1	8	48-55	
关节设定位置	double	6	48	56 - 103	6 关节各自设定的目标位置
预留	double	1	8	104-111	
关节速度	double	6	48	112 - 159	6 关节各自转动速度
预留	double	1	8	160-167	
关节加速度	double	6	48	168 - 215	6 关节各自转动加速度
预留	double	1	8	216-223	
关节电流	double	6	48	224 - 271	6 关节各电机电流
预留	double	1	8	272-279	
关节温度	double	6	48	280 - 327	6 关节各电机温度
预留	double	1	8	328-335	

驱动器温度	double	6	48	336 - 383	6 关节各驱动器温度
预留	double	1	8	384-391	
笛卡尔实际位置	double	6	48	392 - 439	末端笛卡尔空间实际位姿
笛卡尔设定位置	double	6	48	440 - 487	末端笛卡尔空间设定目标位姿
笛卡尔速度	double	6	48	488 - 535	末端笛卡尔空间速度
笛卡尔加速度	double	6	48	536 - 583	末端笛卡尔空间加速度
工具外力	double	6	48	584 - 631	工具坐标系下末端工具所受外力
模拟输入	double	1	8	632 - 639	模拟 IO 输入
模拟输出	double	1	8	640 - 647	模拟 IO 输出
逆运动学关节位姿	double	6	48	648 - 695	逆运动学求解的各关节位置
预留	double	1	8	696-703	
正运动学末端位姿	double	6	48	704 - 751	正运动学求解的末端位姿
数字输入	bool	32	32	752 - 783	控制柜数字输入（1-10 预留，11-14 系统 IO，15-22 外部输入，23-32 预留）
数字输出	bool	32	32	784 - 815	控制柜数字输出（1-2 预留，3-8 系统 IO，9-16 外部输出，17-32 预留）
工具 IO 输入	bool	8	8	816 - 823	工具数字输入（1-2 末端输入，3-8 预留）
工具 IO 输出	bool	8	8	824 - 831	工具数字输出（1-2 末端输出，3-8 预留）
工具按键状态	bool	4	4	832 - 835	末端按钮是否按下
碰撞	bool	1	1	836	机器人是否发生碰撞
碰撞轴	char	1	1	837	发生碰撞的轴
急停	bool	1	1	838	急停是否按下
从站就绪	bool	6	6	839 - 844	各从站是否准备就绪
预留	bool	1	1	845	
机器人状态	char	1	1	846	机器人状态，百位数字表示 Home 状态，1 表示当前处于 Home 位置，个位数据含义如下：： Unconnected = 0, Power = 1, Initial = 2, Disable = 3, Enable = 4,

					Standstill = 5, Running = 6, Stopping = 7, Estop = 8, Error = 9, Pause = 10
虚拟 IO	bool	100	100	847 - 946	虚拟 IO
预留	bool	5	5	947 - 951	

Modbus TCP:

机械臂在上电状态下会作为 modbus 服务器监听外部 modbus 客户端的连接。

机械臂 modbus 配置:

线圈 (Coils): 读 (功能码: 1) 写 (功能码: 5)

Addr	节点说明	读写权限
1	锁定状态	R
2	程序运行状态	R
3	空闲状态	R
4	暂停状态	R
5	程序结束 (上升沿)	R
6	错误状态	R
7	控制柜 IO 输出, io_out[1]	R/W
8	控制柜 IO 输出, io_out[2]	R/W
9	控制柜 IO 输出, io_out[3]	R/W
10	控制柜 IO 输出, io_out[4]	R/W
11	控制柜 IO 输出, io_out[5]	R/W
12	控制柜 IO 输出, io_out[6]	R/W
13	控制柜 IO 输出, io_out[7]	R/W
14	控制柜 IO 输出, io_out[8]	R/W
15-32	预留	/
33	末端 IO 输出, tool_out[1]	R/W
34	末端 IO 输出, tool_out[2]	R/W
35-50	预留	/
51	机器人锁定	R
52	启动程序	R
53	暂停程序	R
54	停止程序	R
55-56	预留	/
57	控制柜 IO 输入, IO_in[1]	R
58	控制柜 IO 输入, IO_in[2]	R
59	控制柜 IO 输入, IO_in[3]	R
60	控制柜 IO 输入, IO_in[4]	R
61	控制柜 IO 输入, IO_in[5]	R
62	控制柜 IO 输入, IO_in[6]	R
63	控制柜 IO 输入, IO_in[7]	R
64	控制柜 IO 输入, IO_in[8]	R
65-72	预留	/
73	末端 IO 输入, tool_in[1]	R
74	末端 IO 输入, tool_in[2]	R
75-80	预留	/
81	待机状态	R

82	运行状态	R
83	停止中	R
84	暂停状态	R
85	当前程序运行完成	R
86	机器人报警	R
87	手动模式	R
88	自动模式	R
89	机器人在作业原点	R
90	急停按下	R
91	机器人上电启动（上升沿触发）	W
92	机器人断电（上升沿触发）	W
93	机器人关机（断电状态下，上升沿触发）	W
94	触发当前加载程序（上升沿触发）	W
95	暂停程序（上升沿触发）	W
96	继续运行程序（上升沿触发）	W
97	停止程序（上升沿触发）	W
98	减速（上升沿触发）	W
99	错误复位（上升沿触发）	W
100	预留	/
101 - 200	虚拟 IO	R/W

Register IN: 读（功能码：3）

Addr	节点说明
201	1 关节位置，低 16 位
202	1 关节位置，高 16 位
203	2 关节位置，低 16 位
204	2 关节位置，高 16 位
205	3 关节位置，低 16 位
206	3 关节位置，高 16 位
207	4 关节位置，低 16 位
208	4 关节位置，高 16 位
209	5 关节位置，低 16 位
210	5 关节位置，高 16 位
211	6 关节位置，低 16 位
212	6 关节位置，高 16 位
213	预留
214	预留
215	基坐标 x 位置，低 16 位
216	基坐标 x 位置，高 16 位
217	基坐标 y 位置，低 16 位
218	基坐标 y 位置，高 16 位
219	基坐标 z 位置，低 16 位
220	基坐标 z 位置，高 16 位

221	基坐标 rx 位置, 低 16 位
222	基坐标 rx 位置, 高 16 位
223	基坐标 ry 位置, 低 16 位
224	基坐标 ry 位置, 高 16 位
225	基坐标 rz 位置, 低 16 位
226	基坐标 rz 位置, 高 16 位
227	机器人状态, 百位数字表示 Home 状态, 1 表示当前处于 Home 位置, 个位数据含义如下: 机器人状态, 数据含义: Unconnected = 0, Power = 1, Initial = 2, Disable = 3, Enable = 4, Standstill = 5, Running = 6, Stopping = 7, Estop = 8, Error = 9, Pause = 10

注: RegisterIN 每个地址读取 16bits, 关节位置及末端位置使用 float (32bits) 表示, 使用 2 个 modbus 节点表示 1 个位置信息。以下是使用 2 个读取到的 16 位 short 类型还原成 32 位 float 类型的参考代码:

```

unsigned short Short1; // decimal part 低 16 位
short Short2;         // integer part 高 16 位
// merge two parts into float
Int My32Int = (Short2 << 16) | Short1;
float MyFloat;
memcpy(&MyFloat, &My32Int, 32);

```

MyFloat 为还原后的实际值

6 脚本函数说明

6.1 简介

此脚本函数手册适用于新松协作机器人编程使用。



注意

- 脚本函数名不能使用如下划线、标点符号等特殊字符作为起始字符。

6.2 数据类型和变量

6.2.1 内置数据类型

算术类型：系统内置算术类型包括整型（int）和浮点型（float）数据，其在系统内部分别用 32bit 内存空间进行表示。系统会自动推断算术类型常量的类型（依据常量是否包含小数点），系统变量在定义的时候根据初始化内容来确定类型。

字符串类型 string：系统内部只支持字符串类型的常量，例如“siasun robot”，字符串统一用‘\0’结尾。

位姿 pose：位姿是一个长度为 6 的 float 类型数组。系统支持位姿类型的常量和变量。

数组类型 array：系统支持 int、float 两种类型的内置数组类型。

6.2.2 系统常量

一个形如 42 的值被称作字面值常量。每个字面值常量都对应一种数据类型，字面值常量的形式和值决定了它的数据类型和在脚本中的使用方式。系统支持的常量如下：

基本类型常量：

算术类型常量、True、False

算术类型常量通常指的是整型或者浮点类型的常量，该类型常量可以参与关系运算、算术运算、逻辑运算，并可以为基本类型变量进行赋值和初始化，亦可作为函数的实参进行参数传递。

True 在系统中代指算数类型常量 1。

False 在系统中代指算数类型常量 0。

列表常量:

列表是一组常量的集合,列表的长度任意,列表中只能是算术类型常量。

例如: [680.760, 185.767, 372.607, -130.145, 33.706, 154.567]

列表常量在系统中的主要功能是在数组定义的时候对数组进行初始化(一般用于数组定义表达式),也可以作为函数的实参进行参数传递,或者对相同长度的数组进行赋值操作。

字符串常量:

字符串常量是字符的集合,以 '\0' 结尾,其实际长度大小是常量中字符个数加一。例如,“siasun robot”常量实际包含的字符个数是 13。字符串常量可用于字符数组变量的定义,函数参数传递。

特殊常量:

[POSE]是一个特殊的常量,各元素值为 0,用于 pose 类型变量定义初始化或参数赋值。

6.2.3 变量及变量的定义

变量在系统中通常用一个唯一的标识符来表示(id)。

基本类型变量

a = 123;	# a 整型变量,初始化为 123
b = 11.23;	# b 浮点型变量,初始化为 11.23
c = True;	# c 整型变量,初始化为 1
d = False;	# d 整型变量,初始化为 0

pose 类型变量

pose 类型是长度为 6 的浮点型数组,用来表示机械臂关节的姿态。

p = [POSE];	# pose 类型变量,初始化为全 0
p = [1, 2, 3, 4, 5, 6];	# pose 类型变量,初始化[1,2,3,4,5,6]
p = [1, 2];	# pose 类型变量,初始化[1,2,0,0,0,0]



注意

- 使用 [] 定义变量后，该变量即为 POSE 类型，长度为 6，长度不因初始定义列表的长度而改变，使用时请注意。

字符串类型变量

```
s = "siasun";
```

```
# 字符串类型的变量，长度 7，末尾自动填充 '\0'
```

数组类型变量

```
a1 = Array(8, INT);          # 整型数组变量，长度 8，初始化 0
```

```
a2 = Array(8, FLO);          # 浮点数组变量，长度 8，初始化 0.0
```

```
a3 = Array(7, INT, [1,2,3,4,5]);
```

```
# 整型数组变量，长度 7，初始化 [1,2,3,4,5,0,0]
```

```
a4 = Array(3, INT, [1,2,3,4,5]);
```

```
# 整型数组变量，长度 3，初始化 [1,2,3]
```

```
a5 = Array(4, FLO, [1,2,3]);
```

```
# 浮点数组变量，长度 4，初始化 [1.0,2.0,3.0, 0.0]
```

```
a6 = Array(a1);              # 拷贝复制 a1
```

6.3 表达式

表达式在系统中指的是有返回值的运算集合，是一个递归的定义。

在此用 exp 表示表达式，用 op 表示运算符。

6.3.1 算术运算表达式

系统支持 加 (+) 减 (-) 乘 (*) 除 (/) 四类算数运算，算数运算表达式返回算数类型常量。

6.3.2 关系运算表达式

系统支持 > < >= <= == != 类型的关系运算，关系运算表达式返回常量 0 或 1。

6.3.3 逻辑运算表达式

系统支持 `and` 、 `or` 、 `not` 类型的逻辑运算。

`exp1 and exp2`, 若 `exp1 != 0` 而且 `exp2 != 0`, 那么表达式返回 1, 否则返回 0

`exp1 or exp2`, 若 `exp1 != 0` 或者 `exp2 != 0`, 那么表达式返回 1, 否则返回 0

`not exp1`, 若 `exp1 != 0`, 那么表达式返回 0, 否则返回 1

6.3.4 赋值表达式

变量名 (id) = exp

赋值表达式的左值是一个变量的表示符, 只有在变量已经定义过的情况下该表达式才是一个赋值表达式, 若变量没有定义, 那么就是一个变量定义语句。

6.3.5 函数调用表达式

函数名 (实参列表);

6.4 语句

6.4.1 循环语句 while

```
while (exp):
```

```
    ...
```

```
end
```

6.4.2 break 语句

break 语句:

```
    break;
```

跳出当前的 `while` 循环

例:

```
i = 1;
```

```
while( i < 10 ):
```

```
i = i + 1;

if i > 7:

    break;

end

end
```

6.4.3 函数定义

```
def fun( 参数列表 ):

    语句列表;
    ...

    return exp;

end
```

例：def fun1(pose:p, array:a, int:c, string:str):

```
    ...

end
```

参数类型：

array: 数组类型

string: 字符串类型

int: 整型

float: 浮点型

pose: pose 类型

返回值类型：

int: 整型

float: 浮点型

**注意**

- 自定义函数仅供数学运算使用，内部不支持运动函数及 print 函数。

6.4.4 return 语句

```
return exp;
```

return 语句只能用在函数内部，用来返回函数的运算结果。

return 语句返回的数据类型决定了函数的返回值。若函数中没有包含 return 语句，那么函数不返回任何值。

6.4.5 条件控制语句

形式一：

```
if (exp):  
    ...  
  
elif (exp):  
    ...  
  
else:  
    ...  
  
end
```

形式二：

```
if (exp):  
    ...  
  
end
```

形式三：

```
if (exp):  
    ...
```

```
else:

    ...

end
```

同大多数语言一样，系统支持 if 条件控制语句，当 if / elif 中的判定条件表达式大于 0 时，就执行语句块中的内容。

6.5 系统函数说明

6.5.1 移动控制

<code>movej(array : axis, float : v, float : a, float : rad = 0);</code>
函数说明： 该指令控制机械臂从当前状态，按照关节运动的方式移动到目标关节角状态。 参数说明： axis : axis 数组对应 1-6 关节的目标关节角度，单位 deg v : 关节角速度，单位（系统设定速度的百分比%），取值范围（0,100] a : 关节加速度，单位（系统设定加速度的百分比%），取值范围（0,100] rad : 关节融合半径，默认值为 0/-1，表示无融合。当数值大于 0 时表示与下一条运动融合。 返回值： 无

<code>movej_pose(pose : p, float : v, float : a);</code>
函数说明： 该指令控制机械臂从当前状态，按照关节运动的方式移动到末端目标位置。 参数说明： p : pose 数据结构，或者长度为6的float型数组，对应末端的位姿，位置单位mm，姿态以Rx、Ry、Rz表示，单位deg v : 关节角速度，单位（系统设定速度的百分比%），取值范围（0,100] a : 关节加速度，单位（系统设定加速度的百分比%），取值范围（0,100] 返回值： 无

<code>movel(pose : p, float : v, float : a, float : rad = 0);</code>
函数说明： 该指令控制机械臂末端从当前状态按照直线路径移动到目标状态。 参数说明：

p : pose 数据结构，或者长度为 6 的 float 型数组，对应末端的位姿，位置单位 mm，姿态以 Rx、Ry、Rz 表示，单位 deg
v : 末端速度，单位 (mm/s)
a : 末端加速度，单位 (mm/s²)
rad : 融合半径 (单位: mm)，默认值为 0/-1，表示无融合。当数值大于 0 时表示与下一条运动融合。
返回值:
无

movec(pose : p1, pose : p2, float : v, float : a, float : rad = 0);

函数说明:

该指令控制机械臂做圆弧运动，起始点为当前位姿点，途径 p1 点，终点为 p2 点。

参数说明:

p1 : pose 类型变量，或长度为 6 的 float 型数组，圆弧运动中间点
p2 : pose 类型变量，或长度为 6 的 float 型数组，圆弧运动结束点
v : 末端速度，单位 (mm/s)
a : 末端加速度，单位 (mm/s²)
rad : 融合半径 (单位: mm)，默认值为 0/-1，表示无融合。当数值大于 0 时表示与下一条运动融合。

返回值:

无

tcp_move(pose:pose_offset, float : v ,float : a, float : rad);

函数说明:

该指令控制机械臂沿工具坐标系直线移动一个增量

参数说明:

pose_offset:pose 数据类型，或者长度为 6 的 float 型数组，表示工具坐标系下的位姿偏移量
v: 直线移动的速度，单位 (mm/s)，当 x、y、z 均为 0 时，线速度按比例换算成角速度。
a : 加速度，单位 (mm/s²)，可选填，默认为 500mm/s²
rad : 融合半径，单位 (mm)，可选填，默认为 0/-1，表示无融合

返回值:

无

tcp_move_2p (pose:p1, pose:p2 , float : v, float : a, float : rad);

函数说明:

该指令控制机器人沿工具坐标系直线移动一个增量，增量为 p1 与 p2 点之间的差

参数说明:

p1 : pose 数据类型，或者长度为 6 的 float 型数组，表示工具坐标系下

的位姿偏移量计算点 1
p2 : pose 数据类型，或者长度为 6 的 float 型数组，表示工具坐标系下的位姿偏移量计算点 2
v: 直线移动的速度，单位 (mm/s)，当 x、y、z 均为 0 时，线速度按比例换算成角速度。
a : 加速度，单位 (mm/s²)，可选填，默认为 500mm/s²
rad : 融合半径，单位 (mm)，可选填，默认为 0/-1，表示无融合
返回值：
无

speedj (array : axis_speed);

函数说明：
该指令控制机械臂每个关节按照给定的速度一直运动，函数执行后会直接运行后续指令。运行 speedj 函数后，机械臂会持续运动并忽略后续运动指令，直到接收到 speed_stop()函数后停止。

参数说明：
axis_speed: 每个关节的速度，单位deg/s

返回值：
无

speedl (pose : position_speed);

函数说明：
该指令控制机械臂末端按照给定的速度一直运动，函数执行后会直接运行后续指令。运行 speedl 函数后，机械臂会持续运动并忽略后续运动指令，直到接收到 speed_stop()函数后停止。

参数说明：
position_speed: pose 类型，或者长度为 6 的 float 型数组，末端速度向量，线速度单位 mm/s,角速度单位 deg/s

返回值：
无

speed_stop ();

函数说明：
停止 speedj 及 speedl 函数的运动。

参数说明：
无

返回值：
无

move_camera_offset (pose : p1, pose : p2, pose : offset, float : v, float : a);

函数说明：

该指令应用于相机拍照抓取 参数说明： p1: 示教的拍照点，pose数据类型，或者长度为6的float型数组 p2: 示教的抓取点，pose数据类型，或者长度为6的float型数组 offset: 相机返回的位置补偿，pose 数据类型，或者长度为 6 的 float 型数组 v: 末端速度，单位 mm/s a: 末端加速度，单位 mm/s2。选填，默认为 500mm/s2。 返回值： 无
--

6.5.2网络通信

socket_open(string : ip, int : port); 函数说明： 打开一个以太网的通信链接，如果三秒内没有创建成功，那么链接创建失败。 参数说明： ip: 服务器端的 IP 地址，需要用“”括起来 port: 服务器端的端口号 返回值： 打开成功返回 True ，失败返回 False 示例： socket_open("192.168.67.60",2003);
--

socket_send_string (string : msg_send); 函数说明： 发送字符串给已连接的服务器 参数说明： msg_send: 需要发送的字符串，需要用“”括起来 返回值： True / False 示例： socket_send_string (“send message”); #发送一位字符： s = “XYZ”; socket_send_string (s);
--

socket_send_float (Array : array_send); 函数说明： 发送浮点数数组给已连接的服务器 参数说明：
--

`array_send`：需要发送的浮点数数组，数组中每个数据为 32 位。

返回值：

True / False

示例：

```
a = Array(2,FLO);
a[0]=3.1416;
a[1]=2.71828;
socket_send_float ( a );
```

`socket_send_int ( Array : array_send);`

函数说明：

发送整型数组给已连接的服务器

参数说明：

`array_send`：需要发送的整型数组，数组中每个数据为 32 位。

返回值：

True / False

示例：

```
a = Array(2,INT);
a[0]=3;
a[1]=2;
socket_send_int ( a );
```

`socket_read_string ( string : receive, int : length, int : timeout = 3000 );`

函数说明：

从已连接的服务器接收一定长度的字符串

参数说明：

`receive`：接收信息的字符串，收到的字符串会存在此变量中

`length`：需要接收的字符串长度

`timeout`: 超时时间，单位 ms。不填写时默认为 3000ms。

返回值：

收到的字节数量，当出错或者字符串长度超限时返回-1

示例：

```
a = "ss";
b = "abc";
while( a[0] != b[0] ):
    socket_read_string ( a, 2 );
end
```

`socket_read_str_float ( Array : receive, int : timeout = 3000 );`

函数说明：

从已连接的服务器接收一组字符串，解析成浮点数

参数说明：

receive : 接收信息的数组, 收到的浮点数会存在此数组中
所有数据应当在括号内, 浮点数之间通过“, ”隔开,

示例: (1.23,3.1416,1.0)

注: 字符串长度最大为 100

timeout: 超时时间, 单位 ms。不填写时默认为 3000ms。

返回值:

收到的字节数量, 当出错或者字符串长度超限时返回-1

示例:

```
a = Array(2,FLO);
socket_read_str_float ( a );
printf(a[0]);
printf(a[1]);
```

socket_read_float (Array : receive, int : timeout = 3000);

函数说明:

从已连接的服务器接收一组浮点数

参数说明:

receive : 接收信息的数组, 收到的浮点数会存在此数组中

注: 一个命令中最多可读取 30 个数值, 每个数据为32位

timeout: 超时时间, 单位 ms。不填写时默认为 3000ms。

返回值:

收到的字节数量, 当出错时返回-1

示例:

```
a = Array(2,FLO);
socket_read_float ( a );
printf(a[0]);
printf(a[1]);
```

socket_read_str_int (Array : receive, int : timeout = 3000);

函数说明:

从已连接的服务器接收字符串, 解析成一组整型数

参数说明:

receive : 接收信息的数组, 收到的整型数会存在此数组中

所有数据应当在括号内, 整型数之间通过“, ”隔开,

示例: (1,3,5)

注: 字符串长度最大为 100

timeout: 超时时间, 单位 ms。不填写时默认为 3000ms。

返回值:

收到的字节数量, 当出错或者字符串长度超限时返回-1

示例:

```
a = Array(2,INT);
socket_read_str_int ( a );
print_i(a[0]);
```

```
print_i(a[1]);
```

```
socket_read_int( Array : receive, int : timeout = 3000 );
```

函数说明：
从已连接的服务器接收一组整型数据

参数说明：
receive：接收信息的数组，收到的整型数会存在此数组中
注：一个命令中最多可读取 30 个数值，每个数据32 位。
timeout: 超时时间，单位 ms。不填写时默认为 3000ms。

返回值：
收到的字节数量，当出错时返回-1

示例：
a = Array(2,INT);
socket_read_int (a);
print_i(a[0]);
print_i(a[1]);

```
socket_close( );
```

函数说明：
关闭和服务器端的以太网通信链接

参数说明：
无

返回值：
关闭成功返回True，失败返回 False

6. 5. 3系统函数及外设

```
sleep( int : time );
```

函数说明：
该函数会让程序暂停一定时间

参数说明：
time：需要暂停的时间，单位 ms。

返回值：
无

```
io_out( int : portnum, int : val );
```

函数说明：
该函数可控制控制柜上的 IO 输出口的高低电平

参数说明：
portnum：控制柜上的 IO 输出口序号，范围从 1-8。
val：1 为高电平，0 为低电平。
参数错误时函数不改变 IO 输出。

返回值: 无

<code>io_in(int : portnum);</code>
函数说明: 该函数可读取控制柜上的用户 IO 输入口的高低电平，返回 1 为高电平，0 为低电平。
参数说明: portnum : 控制柜上的 IO 输入口序号，范围从 1-8。
返回值: int，返回 1 为高电平，0 为低电平，-1 表示参数错误。

<code>sys_io_in(int : portnum);</code>
函数说明: 该函数可读取系统 IO 输入，返回 1 为高电平，0 为低电平。
参数说明: portnum : 控制柜上的 IO 输入口序号，范围从 1-6。 1: 机器人锁定 2: 启动程序 3: 暂停程序 4: 停止程序 5: 预留 6: 预留
返回值: int，返回 1 为高电平，0 为低电平，-1 表示参数错误。

<code>io_v_out(int : portnum, int : val);</code>
函数说明: 该函数可控制系统内部虚拟 IO 的高低电平
参数说明: portnum : 虚拟 IO 序号，范围从 1-100。 val : 1 为高电平，0 为低电平。 参数错误时函数不改变 IO 输出。
返回值: 无

<code>io_v_in(int : portnum);</code>
函数说明: 该函数可读取系统内部虚拟 IO 的高低电平，返回 1 为高电平，0 为低电平。
参数说明:

portnum : 虚拟 IO 序号, 范围从 1-100。
返回值: int, 返回 1 为高电平, 0 为低电平, -1 表示参数错误。

tool_io_out(int : portnum, int : val);
函数说明: 该函数可控制机械臂末端的 IO 输出口的高低电平
参数说明: portnum : 机械臂末端的 IO 输出口序号, 范围从 1-2。 val : 1 为高电平, 0 为低电平。 参数错误时函数不改变 IO 输出。
返回值: 无

tool_io_in(int : portnum);
函数说明: 该函数可读取控机械臂末端的 IO 输入口的高低电平, 返回 1 为高电平, 0 为低电平。
参数说明: portnum : 机械臂末端的 IO 输出口序号, 范围从 1-2。
返回值: int, 返回 1 为高电平, 0 为低电平, -1 表示参数错误。

w_reg_bool(int : num, int : val);
函数说明: 该函数可修改内部 bool 寄存器的值
参数说明: num: 内部 bool 寄存器序号, 范围从 1-32。 val : 1 为 true, 0 为 false。 参数错误时函数不改变内部 bool 寄存器数值。
返回值: 无
示例: w_reg_bool(5, 1);

r_reg_bool (int : num);
函数说明: 该函数可读取内部 bool 寄存器的值
参数说明: num: 内部 bool 寄存器序号, 范围从 1-32。
返回值:

int, 返回 1 为 true, 0 为 false, -1 表示参数错误。

示例:

```
print_i( r_reg_bool( 5 ) );
```

w_reg_byte(int : num, int : value);

函数说明:

该函数可修改内部 byte 寄存器的值

参数说明:

num: 内部 byte 寄存器序号, 范围从 1-32。

value : byte值, 取值范围0-255。

返回值:

无

示例:

```
w_reg_byte( 5, 35 );
```

r_reg_byte(int : num);

函数说明:

该函数可读取内部 byte 寄存器的值

参数说明:

num: 内部 byte 寄存器序号, 范围从 1-32。

返回值:

int

示例:

```
a = r_reg_byte( 5 );
print_i( a );
```

w_reg_float (int : num, float : value);

函数说明:

该函数可修改内部 float 寄存器的值

参数说明:

num: 内部 float 寄存器序号, 范围从 1-32。

value: 浮点型数值。

返回值:

无

示例:

```
w_reg_float( 5, 3.14 );
```

r_reg_float (int : num);

函数说明:

该函数可读取内部 float 寄存器的值

参数说明:

num: 内部 float 寄存器序号, 范围从 1-32。

返回值:

float

示例:

```
print_f( r_reg_bool( 5 ) );
```

```
coordinate_user(float : x, float : y, float : z, float : rx, float : ry, float : rz);
```

函数说明:

设置用户坐标系相对于基坐标系的位移, 设置后, 后续运动函数中的位置均使用此工具坐标系。

参数说明:

x, y, z, rx, ry, rz : 用户坐标系相对于基坐标系的位移 (单位: mm, deg)

返回值:

无

```
coordinate_tool(float : x, float : y, float : z, float : rx, float : ry, float : rz,
float: tcpLoad, float loadBiasX, float loadBiasY, float loadBiasZ);
```

函数说明:

设置工具末端相对于法兰面坐标系的位移, 设置后, 后续运动函数中的位置均使用此工具坐标系。

参数说明:

x, y, z, rx, ry, rz : 工具末端相对于法兰面坐标系的位移 (单位: mm, rad)

tcpLoad: 负载质量 (单位: kg), 使用函数时可省略此变量, 省略后系统会使用当前设置的负载质量进行计算。

loadBiasX, loadBiasY, loadBiasZ: 负载质心相对末端偏移 (单位: mm), 使用函数时这 3 个参数可省略, 省略后系统会使用当前设置的负载质心偏移进行计算。

返回值:

无

```
coordinate_clear();
```

函数说明:

取消已设定的坐标系, 后续移动函数使用基坐标。

参数说明:

无

返回值:

无

```
set_tcp_load(float: tcpLoad, float loadBiasX, float loadBiasY, float loadBiasZ);
```

函数说明:

设置末端工具的质量质心

参数说明:

tcpLoad: 负载质量 (单位: kg)

loadBiasX, loadBiasY, loadBiasZ: 负载质心相对末端偏移 (单位: mm), 使用函数时这 3 个参数可省略, 省略后系统会使用当前设置的负载质心偏移进行计算。

返回值:

无

cal_ikine(pose : p, array : res);

函数说明:

计算逆解

参数说明:

p: 需要计算逆解的末端位置

res: 获取的 1-6 轴关节角度会记录在此 Array 变量中。

返回值: 无

cal_fkine(array : axis, pose : res);

函数说明:

计算正解

参数说明:

axis: 需要计算正解的关节角

res: 获取的末端位置会记录在[POSE]变量 p 中。

返回值: 无

get_flange_pose(pose : p);

函数说明:

该函数可获取当前状态下机械臂的末端法兰中心位姿

参数说明:

p: 获取的末端位置会记录在[POSE]变量 p 中。

返回值:

无

get_tcp_pose(pose : p);

函数说明:

该函数可获取当前状态下机械臂工具末端点的位姿

参数说明:

p: 获取的末端位置会记录在[POSE]变量 p 中。

返回值:

无

get_joint_pose(array : axis);
函数说明： 该函数可获取当前状态下机械臂各关节的角度
参数说明： axis：获取的 1-6 轴关节角度会记录在此 Array 变量中。
返回值： 无

sub_prog(string : program_name);
函数说明： 调用子程序
参数说明： program_name：子程序名，使用双引号 “ ” 包围。
返回值： 无
示例： sub_prog(“program1.spf”);

collision_detect(int :status);
函数说明： 利用脚本开关碰撞检测
参数说明： status：1 表示开启碰撞检测，0 表示关闭碰撞检测，其余值不改变当前碰撞检测状态。
返回值： 无

6.5.4调试相关

print_i(int : i);
函数说明： 该函数可在程序栏旁输出参数 i 的值，方便调试使用。 注：函数运行时有 1s 延时，建议仅在调试时使用。
参数说明： i：整型 INT 变量。
返回值： 无
示例： a = -1; print_i(a);

print_f(float : f);

函数说明： 该函数可在程序栏旁输出参数 f 的值，方便调试使用。 注：函数运行时有 1s 延时，建议仅在调试时使用。 参数说明： f：浮点型 FLO 变量。 返回值： 无 示例： f = 6.32; print_f(f);

print_s(string : s); 函数说明： 该函数可在程序栏旁输出参数 s 的值，方便调试使用。 注：函数运行时有 1s 延时，建议仅在调试时使用。 参数说明： s：字符串变量。 返回值： 无 示例： s="siasun"; print_s(s);
--

6. 5. 5Modbus

modbus_read(string : signal_name); 函数说明： 该函数可读取 modbus 节点的数据，返回值为 int 类型。 参数说明： signal_name: modbus 节点名。 返回值： Int 示例： modbus_read(“mbus1”); #在界面中打印节点的值： print_i(modbus_read(“mbus1”));

modbus_write_d(string : signal_name, int : digital_value); 函数说明： 该函数可对 digital 类型的 modbus 节点进行写操作。 参数说明： signal_name: modbus 节点名。 digital_value: 要写入的数值，取值为 0 或 1。

返回值:

无

示例:

```
modbus_write_d("mbus1", 1);
```

```
modbus_write_r(string : signal_name, int : register_value);
```

函数说明:

该函数可对 register 类型的 modbus 节点进行写操作。

参数说明:

signal_name: modbus 节点名。

register_value: 要写入的数值, 取值范围 -32768~32767

返回值:

无

示例:

```
modbus_write_r("mbus1", 255);
```

```
modbus_set_frequency(string : signal_name, int : frequency);
```

函数说明:

该函数可修改 modbus 节点的刷新频率, 默认频率为 10Hz。

参数说明:

signal_name: modbus 节点名。

frequency: 频率值, 取值范围: 1~100Hz。

返回值:

无

示例:

```
modbus_set_frequency ("mbus1", 20);
```

```
modbus_add_signal(string : IP, int : slave_number, int : signal_address, int :  
signal_type, string : signal_name);
```

函数说明:

该函数可添加 modbus 节点, 也可在系统设置中手动添加。

参数说明:

IP: Modbus 服务器的 IP 地址, 用字符串表示

slave_number: slave 编号

signal_address: 节点地址偏移 Addr

signal_type: 0: Read Discrete Input (1x)

1: Read Write Coils (0x)

2: Read Input Register (3x)

3: Read Write Holding Register (4x)

signal_name: modbus 节点名, 用字符串表示

返回值:

无

示例：

```
modbus_add_signal ("127.0.0.1", 1, 3, 0, "mbus1");
```

```
modbus_delete_signal(string : signal_name);
```

函数说明：

该函数可删除 modbus 节点，也可在系统设置中删除 modbus 节点。

参数说明：

signal_name: modbus 节点名。

返回值：

无

示例：

```
modbus_delete_signal ("mbus1");
```

中科新松有限公司 **SIASUN Co., Ltd**

地址：上海市浦东金桥出口加工区金藏路 257 号

电话：+086-021-50870608

传真：+086-021-63631363

邮编：201206

<http://www.siasun.com>

Add: NO.257 Jinzang Rd.

Pudong New District, Shanghai, China

Tel: +086-021-50870608

Fax: +086-021-63631363

Zip Code: 201206

<http://www.siasun.com>