DobotSCStudio

用户手册（CR 机器人）

文档版本：V1.3.0

发布日期：2020.11.19

深圳市越疆科技有限公司

注意事项

版权所有 © 越疆科技有限公司2020。 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、复制本文档内容的部分或全部，并不得以任何形式传播。

免责声明

在法律允许的最大范围内，本手册所描述的产品（含其硬件、软件、固件等）均“按照现状”提供，可能存在瑕疵、错误或故障，越疆不提供任何形式的明示或默示保证，包括但不限于适销性、质量满意度、适合特定目的、不侵犯第三方权利等保证；亦不对使用本手册或使用本公司产品导致的任何特殊、附带、偶然或间接的损害进行赔偿。

在使用本产品前详细阅读本使用手册及网上发布的相关技术文档并了解相关信息，确保在充分了解机械臂及其相关知识的前提下使用机械臂。越疆建议您在专业人员的指导下使用本手册。该手册所包含的所有安全方面的信息都不得视为Dobot的保证，即便遵循本手册及相关说明，使用过程中造成的危害或损失依然有可能发生。

本产品的使用者有责任确保遵循相关国家的切实可行的法律法规，确保在越疆机械臂的使用中不存在任何重大危险。

越疆科技有限公司

地址：深圳市南山区留仙大道3370号南山智园崇文区2号楼9-10楼

网址：cn.dobot.cc

前言

目的

本手册介绍了机器人控制软件DobotSCStudio的功能和使用方法，方便用户了解和使用Dobot机器人。

读者对象

本手册适用于：

- 客户
- 销售工程师
- 安装调测工程师
- 技术支持工程师

修订记录

时间	修订记录
2020/11/19	第三次发布 增加连接虚拟控制器说明、夹爪安装章节
2020/09/27	第二次发布 根据最新软件刷新界面，增加快速连接、安全设置等功能说明
2020/06/03	第一次发布

符号约定

在本手册中可能出现下列标志，它们所代表的含义如下。

符号	说明
⚠危险	表示有高度潜在危险，如果不能避免，会导致人员死亡或严重伤害
⚠警告	表示有中度或低度潜在危害，如果不能避免，可能导致人员轻微伤害、机械臂毁坏等情况
⚠注意	表示有潜在风险，如果忽视这些文本，可能导致机械臂损坏、数据丢失或不可预知的结果
📖说明	表示是正文的附加信息，是对正文的强调和补充

目录

注意事项 i
前言 ii
1. 简介 1
1.1 主界面说明 1
2. 快速连接 4
2.1 直接连接 4
2.2 通过 WiFi 连接 6
3. 功能说明 8
3.1 参数设置 8
3.1.1 用户坐标系设置 8
3.1.2 工具坐标系设置 10
3.1.3 回零设置 12
3.1.4 I/O 监控 13
3.1.5 控制器设置 15
3.1.6 远程控制 16
3.1.7 机器人参数设置 19
3.1.8 安全设置 24
3.2 工具配置 28
3.2.1 基础配置 29
3.2.2 插件信息 29
3.2.3 日志 30
3.2.4 网络配置 30
3.2.5 机器人状态 32
3.2.6 调试工具 33
3.2.7 虚拟仿真 33
3.2.8 WiFi 设置 34
3.3 编程 34
3.3.1 工程说明 34
3.3.2 编程面板说明 35
3.3.3 编程说明 36
3.4 使能 45
3.5 设置全局速率 45
3.6 报警说明 46
3.7 末端负载 47
4. 编程语言 48
4.1 算术运算符 48
4.2 关系运算符 48
4.3 逻辑运算符 48
4.4 一般关键字 49
4.5 一般符号 49
4.6 过程控制指令 49
4.7 全局变量 49

4.8 运动指令 50
4.9 运动参数设置指令 56
4.10 六维传感器运动指令 58
4.11 输入/输出指令 60
4.12 程序管理指令 62
4.13 获取位姿指令 65
4.14 TCP 66
4.15 UDP 70
4.16 Modbus 73
4.16.1 Modbus 寄存器说明 73
4.16.2 指令说明 75

1. 简介

DobotSCStudio是越疆科技退出的一款工业机器人编程平台，适用于越疆全系列的工业机器人（SA/SR/CR系列），界面友好，支持用户二次开发。DobotSCStudio还提供多种机械结构的运动学算法，内置虚拟仿真环境，适用于各种工艺应用。

1.1 主界面说明

DobotSCStudio主界面如图 1.1所示，其界面详细说明如表 1.1所示。

图 1.1 主界面

表 1.1 DobotSCStudio 面板说明

编号	说明
1	工程 用户可导入工程，并对工程进行调试，运行等操作
2	点动 在不同坐标系下控制机械臂的运动，仅在手动模式下有效 若在关节坐标系下点动：从上往下表示点动J1、J2、J3、J4、J5、J6 若在笛卡尔坐标系下点动：从上往下表示点动X、Y、Z、Rx、Ry、Rz
3	系统 用户可在该页面进行系统设置，如网络设置，机器人参数设置，坐标系设置，工艺包设置等

编号	说明
4	系统菜单栏
5	单击图标可切换手动模式和自动模式 • ：手动模式。点动机械臂、编程或设置参数等 • ：自动模式。监控运动轨迹、I/O端口或调试运行程序等
6	• 手动模式下仅表示机械臂电机使能状态 • 自动模式下单击此按钮控制电机使能
7	机械臂报警信息查询 当机械臂有报警时，该图标变为红色 可在操作面板查看具体的报警信息并将控制柜切换至手动模式后清除报警
8	设置机械臂全局速率比例
9	IP设置以及检查更新 • 通过网线连接控制柜和电脑后，用户需在“IP设置”页面勾选“真实”，然后选择控制柜IP地址，以便连接DobotSCStudio • 如果用户无设备使用时，可在“IP设置”页面勾选“虚拟”，通过连接虚拟控制柜来体验DobotSCStudio
10	急停开关 机械臂运行过程中出现突发情况时可按下急停开关，使控制柜伺服驱动断电，机械臂紧急停止
11	用户模式选择 • 观察员：查看全局变量、I/O状态、机器姿态、位置信息、报警信息等 • 操作人员：在监控人员权限的基础上，还可基于现有的程序脚本文件操作，无需编程 • 程序员：在操作人员权限的基础上，还可编程、示教 • 管理员：可设置或修改任何参数 根据使用者的角色，双击图标选择用户模式 默认密码：admin，用户可在管理员模式下进入“工具配置 > 基础配置 > 用户模式”页面下修改密码
12	交互窗口

编号	说明
13	显示当前的运行模式 运行模式包括：I/O、Modbus、SCStudio

版权所有 © 越疆科技有限公司

2. 快速连接

本节仅描述如何连接DobotSCStudio与机械臂。以CR5机械臂为例，机械臂与控制柜之间的连接请参见《Dobot CR5硬件使用手册》。

2.1 直接连接

DobotSCStudio可与控制柜通过网线直接连接进行通信。此时控制柜的IP地址需与PC机的IP地址在同一网段。

通过网线连接时，控制柜默认的IP地址为192.168.5.1，用户需修改PC机的IP地址，使其与控制柜的IP地址在同一网段。

说明

本示例以Win7操作系统为例，请根据实际操作系统修改PC机的IP地址。

步骤 1 将网线一端连接至控制柜的LAN接口，另一端连接至PC。

步骤 2 在PC机上单击“开始 > 控制面板”菜单选择“网络和共享中心”。

弹出“网络和共享中心”窗口。

步骤 3 在“网络和共享中心”窗口单击“本地连接”。

弹出“本地连接状态”界面。

步骤 4 在“本地连接状态”界面单击“属性”。

弹出“本地连接属性”界面。

步骤 5 在“本地连接属性”界面双击“Internet协议版本4（TCP/IPv4）”。

弹出“Internet协议版本4（TCP/IPv4）属性”界面。

步骤 6 在“Internet协议版本4（TCP/IPv4）属性”界面选择“使用下面的IP地址”，修改PC机的IP地址、子网掩码以及默认网关，如图 2.1所示。

可将PC机的IP地址修改为与控制柜同一网段未被占用的任意IP地址，其子网掩码和默认网关与控制柜的一致。

注意

如果PC机和控制柜通过网线直接连接，则不需设置默认网关。

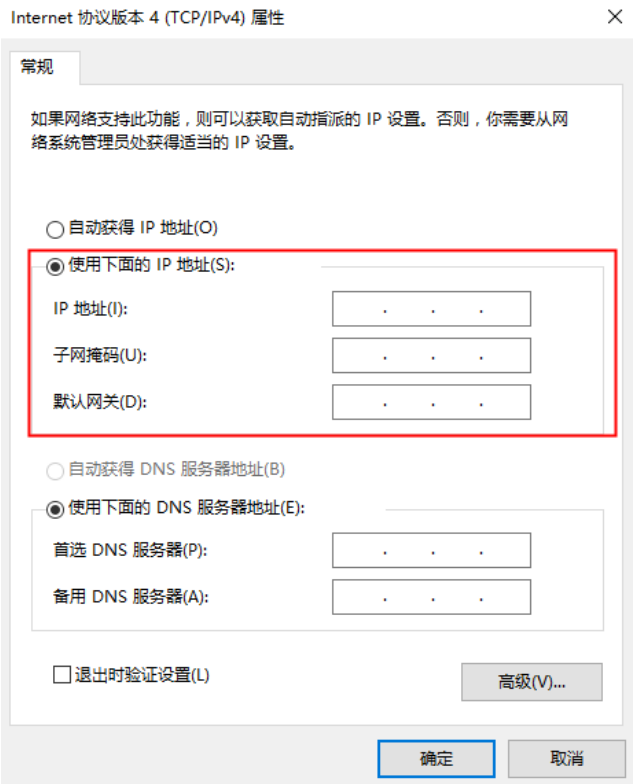


图 2.1 IP 地址修改示意图

步骤 7 单击“确定”。

步骤 8 在DobotSCStudio界面的右上方单击“ > IP设置”，选择控制柜的IP地址并单击“确认”。

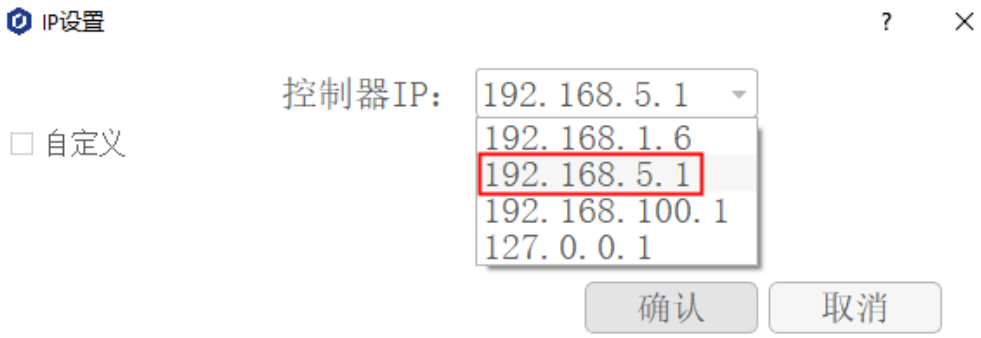


图 2.2 IP 设置

如果无法找到控制柜的IP地址，则需勾选自定义，添加控制柜的IP地址后再单击“确认”。

图 2.3 添加控制柜 IP 地址

连接成功后，DobotSCStudio会显示如图 2.4页面。

图 2.4 连接成功后显示页面

2.2 通过 WiFi 连接

前提条件

- 控制柜已连接WiFi模块。
- PC机支持WiFi连接

操作步骤

步骤 1 在PC机搜索并连接WiFi，WiFi名前缀为“Dobot_WIFI_XXX”。其中XXX为位于机械臂底座上的机械臂编号。初始WiFi密码为：1234567890。

用户可在“工具配置 ＞WiFi设置”修改WiFi密码，重新启动控制柜后生效。

步骤 2 在DobotSCStudio界面的右上方单击“ ＞ IP设置”，选择控制柜的IP地址并单击“确认”。

通过WiFi连接时，控制柜的IP地址为192.168.1.6。

图 2.5 IP 设置

连接成功后，DobotSCStudio会显示如图 2.6页面。

图 2.6 连接成功后显示页面

3. 功能说明

3.1 参数设置

机械臂示教和再现前，需对机械臂进行一系列设置，包括运动参数设置、用户模式、工艺包设置等。

3.1.1 用户坐标系设置

当工件的位置发生变化或机械臂的运行程序需要在多个同类型的加工系统中重复使用时，此时需要设置用户坐标系，使所有路径都跟随用户坐标同步更新，极大的简化了示教编程。当前系统支持10个用户坐标系，用户坐标系**0**为基坐标系，不可更改。

⚠注意

建立用户坐标系时，请确保参考坐标系为默认坐标系，即增加用户坐标系时DobotSCStudio界面的用户坐标系图标为用户坐标系： 0 。

- 点：末端TCP移动到任意一点A来确定坐标系原点，根据默认的工具坐标系方向来确定该用户坐标系方向。

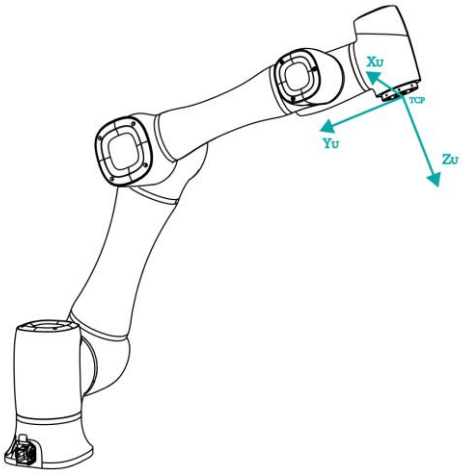


图 3.1 点用户坐标系确定方法

- 线：任意两点A、B确定确认一条直线。从A点到B点方向确定Y轴正方向。根据A点工具坐标系的ZT轴投射到直线垂直面的投影来确定ZU轴正方向。再根据右手法则确定X轴正方向。

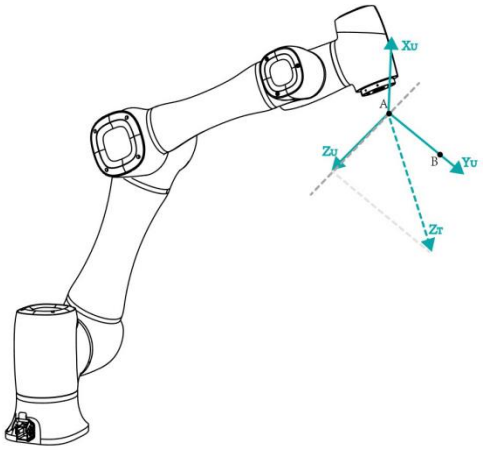


图 3.2 线用户坐标系确定方法

- 面：三点示教法生成：将机械臂移动至任意三点**A**（x1,y1,z1），**B**（x2,y2,z2）和**C**（x3,y3,z3）。其中**A**点作为原点，**AB**两点之间的连线确定用户坐标系X轴正方向，C点沿X轴做垂线确定Y轴正方向，根据右手法则，确定Z轴方向，如所示。

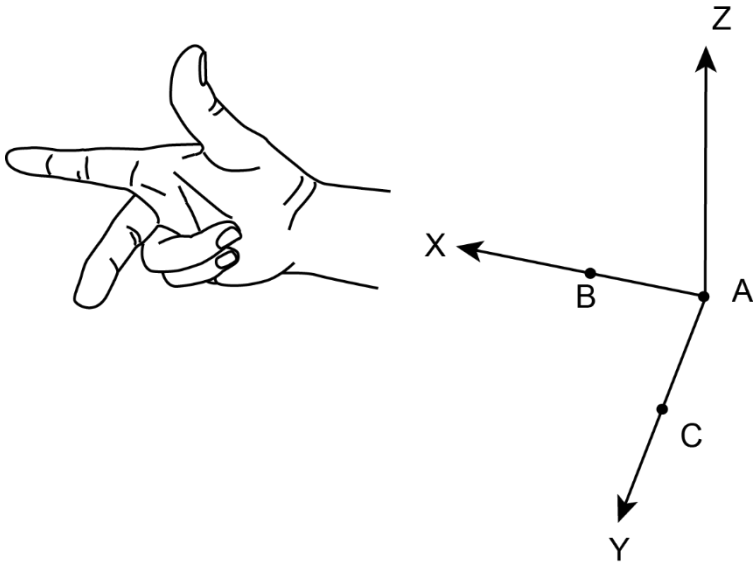


图 3.3 三点示教法

本节根据面的三点示教创建用户坐标系**4**为例进行说明。

前提条件

- 机械臂已上电。
- 控制柜已切换至手动模式。
- 已切换至笛卡尔坐标系。
- 机械臂电机已使能。

操作步骤

步骤 1 单击“ > 参数设置> 坐标系设置 > 用户坐标系”。

进入“用户坐标系”界面，如图 3.4所示。

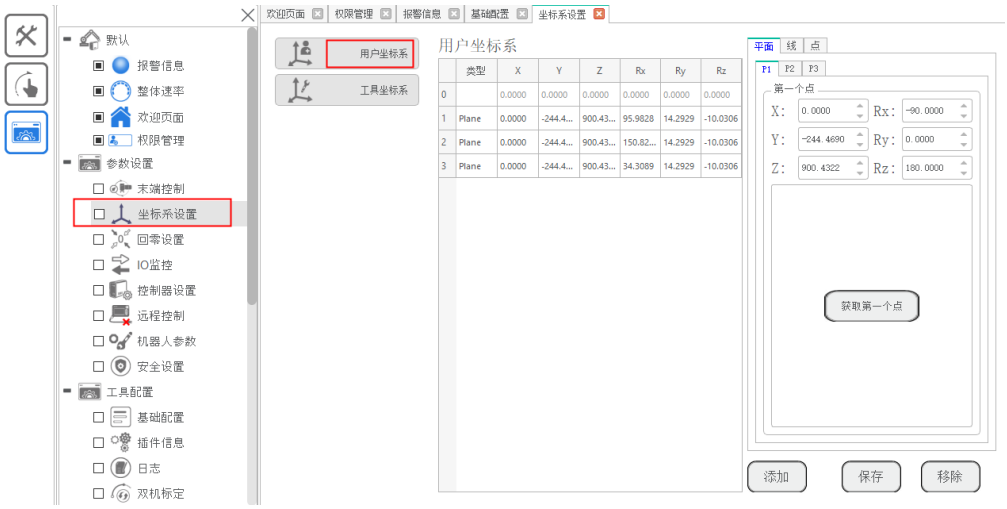


图 3.4 用户坐标系界面

说明

Rx、Ry、Rz为工具中心点TCP绕选定的用户坐标系X、Y、Z轴旋转的角度。

步骤 2 单击“平面”并点动机械臂至任意一点，在“P1”页签单击“获取第一个点”获取第一个点的坐标。

步骤 3 点动机械臂至第二个任意点，在“P2”页签单击“获取第二个点”获取第二个点的坐标。

步骤 4 点动机械臂至第三个任意点，在“P3”页签单击“获取第三个点”获取第三个点的坐标。

步骤 5 单击“添加”和“保存”，用户坐标系4参数自动生成。

步骤 6 在点动界面单击“坐标系选择”， 选择“坐标系：4”。

此时可在用户坐标系**4**下进行示教、编程等操作。

3.1.2 工具坐标系设置

当机械臂末端安装了工具（如焊枪、喷嘴、夹具等）后，为了编程和机械臂运行的需要，此时需要设置工具坐标系。例如，利用多个夹具同时搬运多个工件，可将每个夹具设置为独立的工具坐标系以提高搬运效率。

当前系统支持10个工具坐标，工具坐标系**0**为默认工具坐标系，即不使用工具，不可更改。

注意

建立工具坐标系时，请确保参考坐标系为默认坐标系，即增加工具坐标系时DobotSCStudio界面的工具坐标系图标为 工具坐标系： 0 。

协作工具坐标系采用三点示教法“TCP+ZX”生成：机械臂末端安装工具后，调整工具的位姿，使TCP（Tool Center Point）以三种不同的姿态对准空间中同一点（参考点），获取工具位置偏移。再根据另外三点（**A**、**B**、**C**）获取工具姿态偏移，如图 3.5所示。

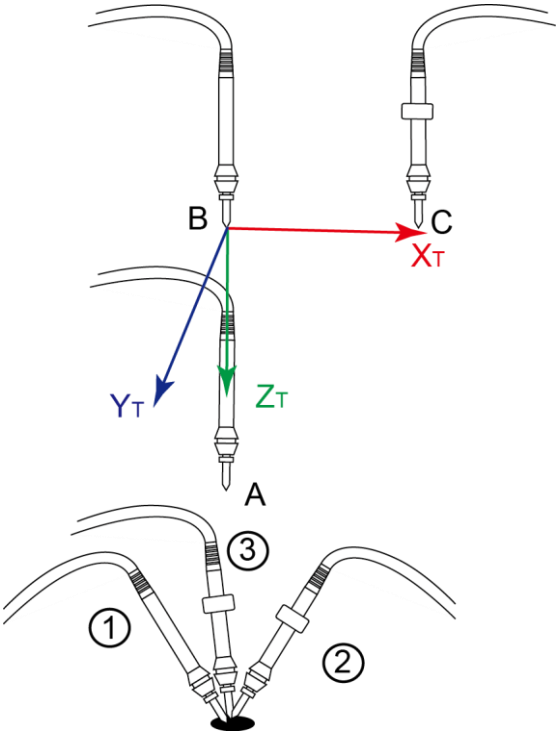


图 3.5 工具坐标系三点示教法“TCP+ZX”

本节以建立工具坐标系**1**为例进行说明。

前提条件

- 机械臂已上电。
- 控制柜已切换至手动模式。
- 机械臂电机已使能。

操作步骤

步骤 1 在机械臂末端安装夹具，本节不做详细说明。

步骤 2 单击“ > 参数设置 > 坐标系设置 > 工具坐标系”。

进入“工具坐标系”界面，如图 3.6所示。

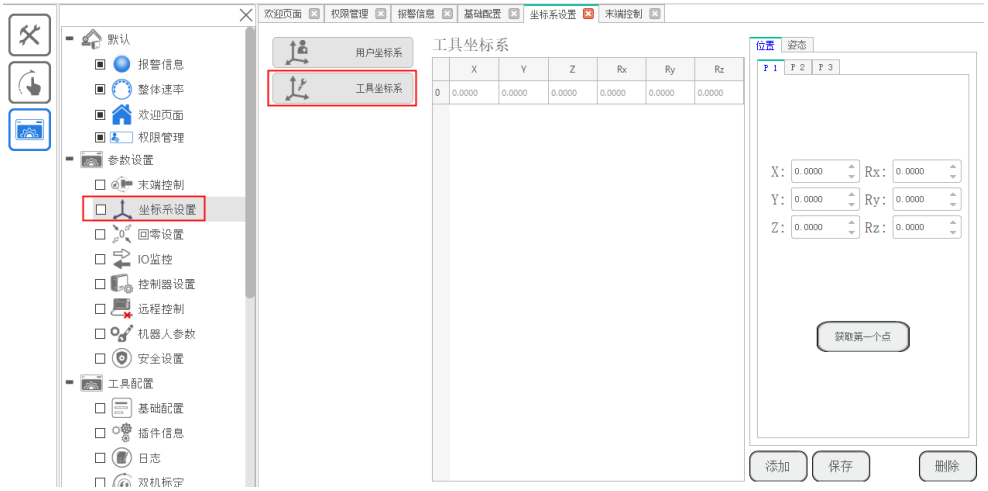


图 3.6 工具坐标系界面

说明

Rx、Ry、Rz为工具中心点TCP绕选定的工具坐标系X、Y、Z轴旋转的角度。

步骤 3 点动机械臂，使TCP（Tool Center Point）以第一种姿态对准参考点（空间中的一点），并在“位置”页签的“P1”区域单击“获取第一个点”获取第一个点的坐标。

步骤 4 点动机械臂，使TCP以第二种姿态对准参考点，并在“位置”页签的“P2”区域单击“获取第二个点”获取第二个点的坐标。

步骤 5 点动机械臂，使TCP以第三种姿态对准参考点，并在“位置”页签的“P3”区域单击“获取第三个点”获取第三个点的坐标。

步骤 6 使能机械臂，并点动机械臂，使TCP以垂直姿态对准参考点，在“姿态”页签的“P1”区域单击“获取第一个点”获取第四个点的坐标（A点）。

步骤 7 正向点动Z轴，使机械臂移动至另一点，在“姿态”页签的“P2”区域单击“获取第二个点”获取第五个点的坐标（B点）。

该步骤确定Z轴正方向。

步骤 8 正向点动X轴，使机械臂移动至C点（不能与A、B两点在同一直线上），在“姿态”页签的“P3”区域单击“获取第三个点”获取第六个点的坐标。

该步骤确定X轴正方向，然后根据右手定则确定Y轴方向。

步骤 9 单击“添加”和“保存”，计算得到工具坐标系**1**的参数。

步骤 10 在点动界面单击“坐标系选择”， 选择“坐标系：1”。

此时可在工具坐标系**1**下进行示教、编程等操作。

3.1.3 回零设置

DobotSCStudio支持将CR系列机器人移动到常用的姿态：出厂姿态、回零姿态、自定义姿态。出厂姿态可缩小机器人占用空间，方便打包运送。回零姿态即零点位置。自定义姿态则根据用户需求自定义常用的姿态，方便快速移动到该位置。

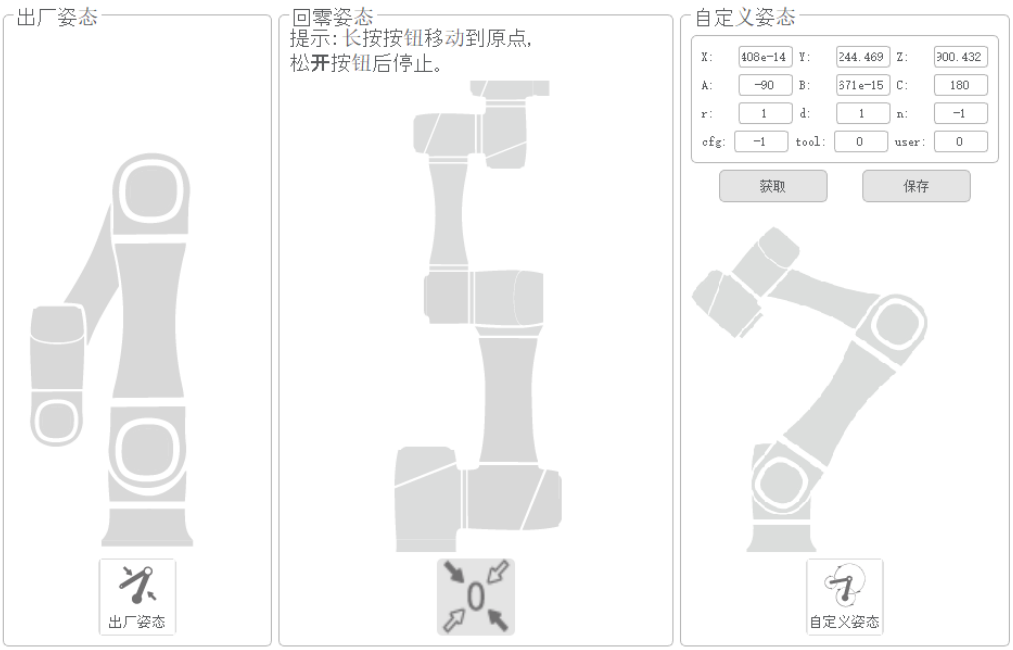


图 3.7 姿态设置

- 出厂姿态：单击 使机械臂移动至出厂位置。
- 回零姿态：长按 使机械臂移动至零点位置。
- 自定义姿态：单击 使机械臂移动至用户自定义的姿态。

移动至自定义姿态前，用户需先在“回零设置”页面自定义姿态：点动机械臂至常用的位置，单击“获取”并单击“保存”。

3.1.4 I/O 监控

该页面可以设置和监控控制柜和本体末端各个I/O状态。单击“ > 参数设置 >IO 监控”进入I/O监控页面，如图 3.8所示，可在该页面设置数字输出状态、模拟输出方式以及值。

图 3.8 I/O 监控页面

I/O监控具有三个功能：输出、监控以及模拟。

- 输出：在手动模式下可以设置数字输出或模拟输出的值。
- 监控：可以查看输入、输出的真实状态，且在自动模式下安全保护，仅支持查看输入输出状态，不能修改。
- 模拟：在手动模式或自动模式下模拟数字输入状态，方便调试运行程序。如图 3.9 所示。

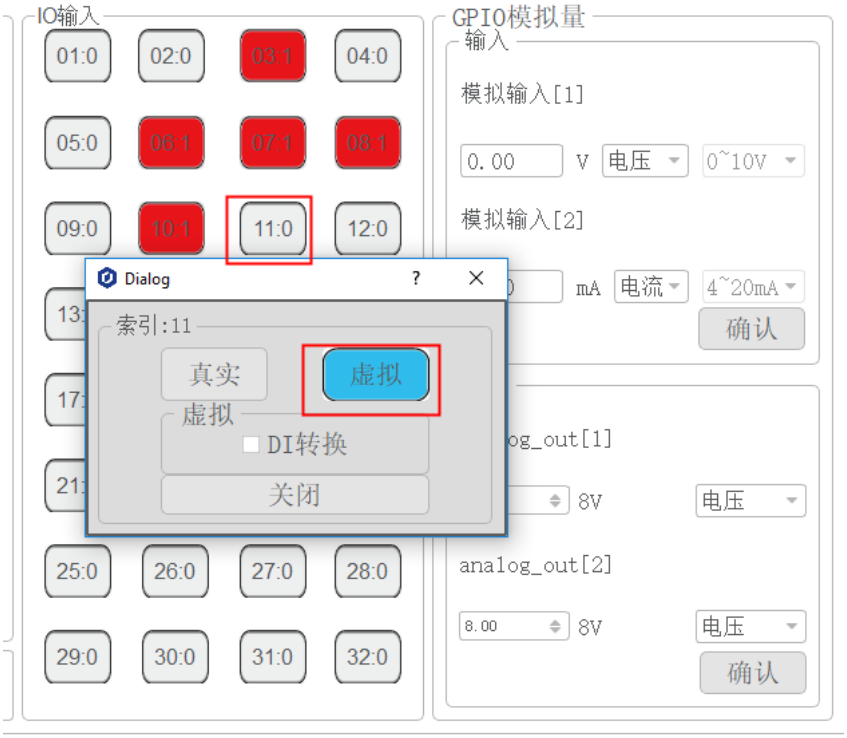


图 3.9 模拟数字输入状态

详细I/O接口说明请参见《Dobot CR5用户手册》。

3.1.5 控制器设置

3.1.5.1 重启

当更新了控制柜固件或者控制柜出现异常情况需要重新启动控制柜时，可通过DobotSCStudio进行重启，即在“参数设置 > 控制器设置 > 重启”页面单击 。

图 3.10 重新启动控制柜

3.1.5.2 更新

当有控制器固件待更新时，可在该页面导入待更新的固件，导入后重新启动控制柜即可。最新固件请联系技术支持获取。

图 3.11 更新固件

3.1.6 远程控制

外部设备可以通过不同的远程控制模式如远程I/O模式、远程Modubs模式下发指令控制机械臂（即对示教好的程序文件进行控制运行）。当需要设置远程控制模式时，需通过DobotSCStudio设置，且机械臂处于下使能状态，否则会触发报警。

⚠注意

- 远程控制模式切换不需重新启动机器人控制系统。
- 无论机器人控制系统处于哪种模式，硬件上的急停开关始终有效。
- 若机械臂在远程控制模式下运行，此时不能切换为其他远程控制模式，需退出当前远程模式后再切换，即先停止运行再切换。
- 若机械臂在上使能状态，则不能触发远程控制，会触发报警。请在下使能状态下触发远程控制。

3.1.6.1 远程 I/O

当远程控制模式为远程I/O时，外部设备可通过远程I/O来控制机器人，如工位预约应用场景。Dobot机器人控制系统特定的I/O接口定义如表 3.1所示。

表 3.1 特定 I/O 接口定义说明

I/O接口	说明
输入（用于外部控制）	
DI 11	清除报警
DI 12	继续运行
DI 13	暂停运行，但不退出远程I/O模式
DI 14	停止运行，退出远程I/O模式
DI 15	进入远程I/O模式，开始运行
DI 16	急停，退出远程I/O模式
输出（用于显示状态）	
DO 13	准备好状态
DO 14	暂停状态
DO 15	报警状态
DO 16	运行状态

⚠注意

所有输入信号为沿触发，低电平变为高电平有效。

前提条件

- 已准备好远程运行的工程文件，且能正确运行。
- 已通过I/O接口连接外部设备，特定I/O接口说明如表 3.1所示。
- 机械臂已上电。

说明

本节不对如何连接外部设备以及如何使用外部设备进行详细说明。

操作步骤

步骤 1 单击“ > 参数设置 > 远程控制”。

进入远程控制页面，如图 3.12所示。

图 3.12 远程控制页面

步骤 2 在“控制模式”区域勾选“IO”，并选择待远程运行的工程，并单击“保存”。

弹出“保存成功，现在远程控制模式为IO！”窗口。

此时，除了急停按钮，其他按键无效。

步骤 3 在外部设备上触发启动信号，此时机械臂会按照选择的工程文件进行运行。

若触发停止信号，则退出远程I/O模式。勾选“SCStudio”并保存进入软件控制模式。

3.1.6.2 远程 Modbus

当远程控制模式为远程Modbus时，外部设备可通过远程Modbus来控制机器人，如工位预约应用场景。Modbus寄存器详细说明请参见4.16.1Modbus寄存器说明。其特定功能如表

3.2所示。

表 3.2 Modbus 寄存器特定功能说明

寄存器（以PLC为例）	寄存器（Dobot机器人控制系统）	说明
线圈寄存器		
00001	0	进入远程Modbus模式，开始运行
00002	1	暂停运行，但不退出远程Modbus模式
00003	2	继续运行
00004	3	停止运行，退出远程Modbus模式
00005	4	急停，退出远程Modbus模式
00006	5	清除报警
触点寄存器		
10001	0	自动退出
10002	1	准备好状态
10003	2	暂停状态
10004	3	运行状态
10005	4	报警状态

前提条件

- 已准备好远程运行的工程文件，且能正确运行。
- 已通过Ethernet接口将外部设备与机械臂连接。可直接连接或通过路由器连接，请根据实际情况选择。

  其中，Dobot机器人控制系统和外部设备的IP地址需在同一网段。Dobot机器人控制系统的IP地址可在“工具配置 > 网络配置”中查询并修改；端口为502，不能修改。
- 机械臂已上电。

  说明

  本节不对如何连接外部设备以及如何使用外部设备进行详细说明。

操作步骤

步骤 1 单击“ > 参数设置 > 远程控制”。

进入远程控制页面，如图 3.13所示。

图 3.13 远程控制页面

步骤 2 在“控制模式”区域勾选“Modbus”，并选择待远程运行的工程，并单击“保存”。

弹出“保存成功，现在远程控制模式为Modbus！”窗口。

此时，除了急停按钮，其他按键无效。

步骤 3 在外部设备上触发启动信号，此时机械臂会按照选择的工程文件进行运行。

若触发停止信号，则退出远程Modbus模式。勾选“SCStudio”并保存进入软件控制模式。

3.1.7 机器人参数设置

机械臂在示教或再现时可设置各坐标系的运动速度和加速度以及再现时门型运动相关参数，设置完成后请单击“保存”。单击“ > 参数设置 > 机器人参数”进入机器人参数界面。

- 关节点动参数：设置点动时各关节轴的最大速度和最大加速度。如图 3.14所示。

图 3.14 点动关节参数

机械臂点动时各关节实际运动速度与最大速度关系如下所示：

- 各关节点动速度=各关节最大速度*全局速率
- 各关节点动加速度=各关节最大加速度*全局速率

说明

- *全局速率可在主界面进行设置，详细请参见3.5 设置全局速率。*
- 指令设置的百分比可在编程时调用速度指令设置。

- 坐标系点动参数：设置点动时各笛卡尔坐标轴的最大速度和最大加速度。如图 3.15 所示。

图 3.15 点动坐标系参数

机械臂点动时各笛卡尔坐标轴实际运动速度与最大运动速度关系如下所示：

- 各笛卡尔坐标轴点动速度=各笛卡尔坐标轴最大速度*全局速率
- 各笛卡尔坐标轴点动加速度=各笛卡尔坐标轴最大加速度速度*全局速率

- 示教再现关节参数：设置再现时各关节轴的最大速度和最大加速度。如图 3.16所示。

图 3.16 示教再现关节参数

机械臂再现时各关节实际运动速度与最大速度关系如下所示：

- 各关节再现速度=各关节最大速度*全局速率*指令设置的百分比
- 各关节再现加速度=各关节最大加速度*全局速率*指令设置的百分比
- 各关节再现加加速度=各关节最大加加速度*全局速率*指令设置的百分比

- 示教再现坐标系参数：设置再现时各笛卡尔坐标轴的最大速度和最大加速度。如图3.17所示。

图 3.17 示教再现关节坐标系参数

机械臂再现时各笛卡尔坐标轴实际运动速度与最大速度关系如下所示：

- 各笛卡尔坐标轴再现速度=各笛卡尔坐标轴最大速度*全局速率*指令设置的百分比
- 各笛卡尔坐标轴再现加速度=各笛卡尔坐标轴最大加速度*全局速率*指令设置的百分比
- 各笛卡尔坐标轴再现加加速度=各笛卡尔坐标轴最大加加速度*全局速率*指令设置的百分比

- 弧形再现参数：再现时若机械运动模式为“Jump”时，需设置起始点抬升高度、结束点下降高度以及抬升最大高度。

  DobotSCStudio支持设置10组Jump相关参数，设置并勾选任一组参数后可在编程时调用，如图 3.18所示。

图 3.18 弧形再现参数

- 机械参数：设置机械臂各轴的参数，例如可设置每个轴为虚拟状态，设置电机方向，转速、最大速度，额定转速，减速比分子，减速比分母等。（该功能由工作人员进行设置，用户无法设置，此处不做详细说明）

图 3.19 机械参数

3.1.8 安全设置

3.1.8.1 抱闸

抱闸是在伺服驱动器处于非运行状态时，防止伺服电机轴运动，使电机保持位置锁定，确保机械的运动部分不会因为自重或外力移动。如果用户需进行关节拖拽操作，可开启抱闸，即在机械臂下使能后，在“参数设置 > 安全设置 > 抱闸”页面开启关节抱闸功能，开启时需用手扶住关节，防止关节移动。

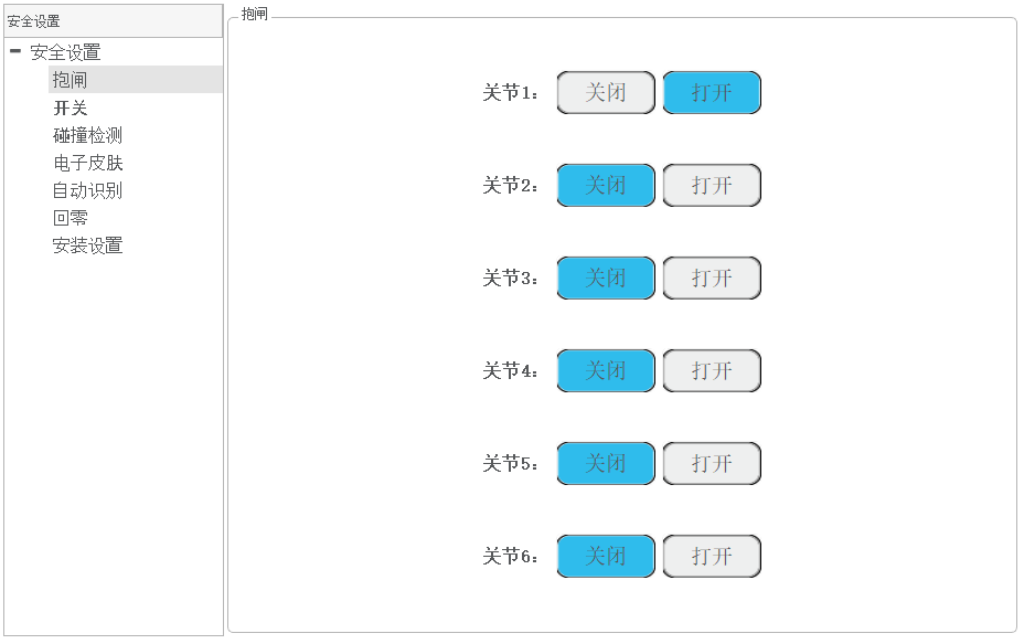


图 3.20 关节抱闸

3.1.8.2 开关

当出现紧急情况按下紧急停止开关后，机械臂本体会断开电源，此时可通过DobotSCStudio开启电源，即在“参数设置 > 安全设置 > 开关”，单击“开机”即可。

也可通过DobotSCStudio控制机械臂本体下电。

图 3.21 控制机械臂上电和下电

3.1.8.3 碰撞检测

碰撞检测主要用于减少碰撞力对机器人本体的影响，避免机器人本体或者外围设备损坏。开启碰撞检测后，当机械臂碰撞到工件或其他障碍物时，机械臂会自动停止运行，以防机器或操作人员碰撞受伤。

用户可在“参数设置 > 安全设置 > 碰撞检测”页面开启碰撞检测并设置碰撞等级。也可同时勾选“开启碰撞后自动拖动功能”，即当机械臂碰撞停住后可手动拖动机械臂到安全的位置。

碰撞等级为安全等级设置，共有1-5个安全等级。等级越高，机械臂碰撞检测后停止所需的力越小。

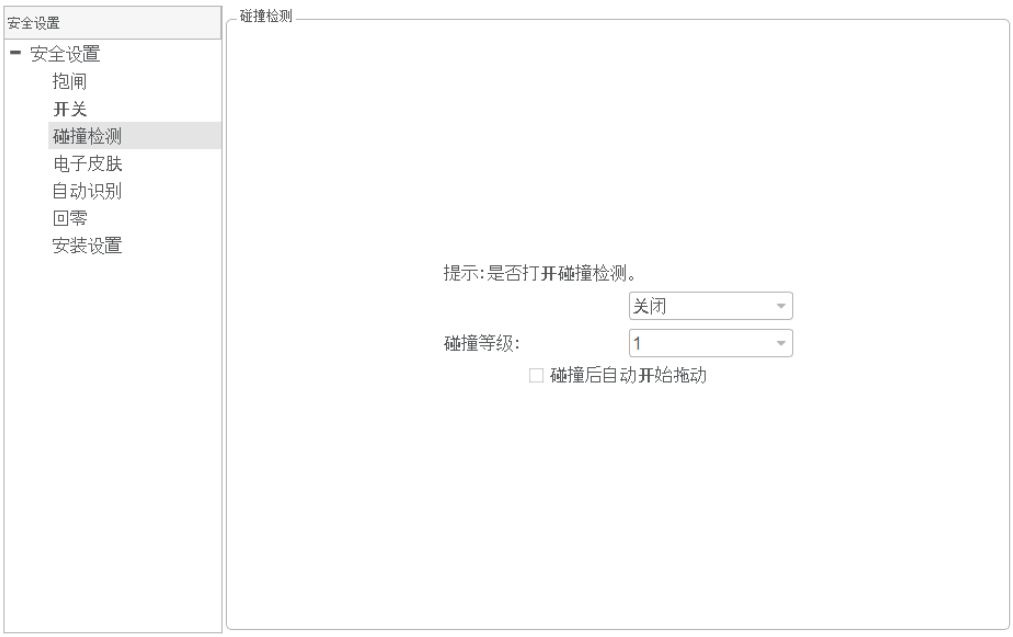


图 3.22 碰撞安全

3.1.8.4 电子皮肤

电子皮肤能够允许机器人实时对障碍物进行反应，帮助机器人避开前进中的障碍，避免了危险碰撞。

用户可在“参数设置 > 安全设置 > 电子皮肤”页面开启电子皮肤并设置遇到障碍物后的反应状态，如遇到障碍物后是暂停运行还是绕过障碍物继续运行。用户还可设置避让和恢复相关参数，如表 3.3所示。

图 3.23 电子皮肤

表 3.3 参数设置

参数	设置
避让速度	遇到障碍物时的避让速度 单位：mm/s 取值范围：1~500，推荐设置为100
避让距离	避让距离，即离障碍物的距离 单位：mm 取值范围：0~200，推荐设置为80
避让加速度	遇到障碍物时的避让加速度 单位：mm/s^2 取值范围：1~50000，推荐设置为1000
恢复速度	绕过障碍物后的恢复速度 单位：mm/s 取值范围：1~500，推荐设置为100
恢复加速度	绕过障碍物后的恢复加速度 单位：mm/s^2 取值范围：1~50000，推荐设置为1000

3.1.8.5 回零

当更换机械臂电机、减速机等传动部件或者与工件发生碰撞等情况下，机械臂的零点位置发生变化，此时需对机械臂进行回零操作。

在手动模式下，对齐机械臂相邻关节的键槽，然后使能机械臂，单击 即可。

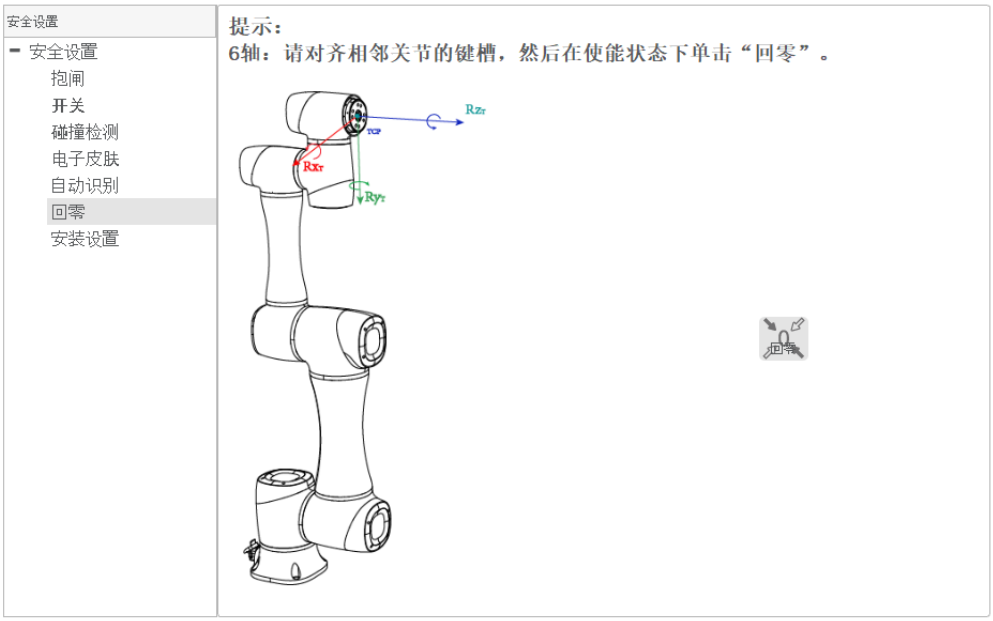


图 3.24 回零

3.1.8.6 安装设置

一般情况下机械臂安装在平稳的台面或者地面上，此种情况无需在此页面进行任何操作。如果机械臂采用吊顶式、壁挂式安装或者呈一定角度安装，则需要在下使能状态下设置旋转角度和倾斜角度。

倾斜角度是指本体在原点位置绕X轴逆时针旋转的角度。

旋转角度是指本体在原点位置绕Z轴逆时针旋转的角度。

图 3.25 安装设置

3.2 工具配置

3.2.1 基础配置

用户可在“工具配置 > 基础配置 > 版本”页面查看软件、控制器、算法等版本信息，也可在“工具配置 > 基础配置 > 语言”页面设置系统语言，当前仅支持中英文两种语言。还可在“工具配置 > 基础配置 > 用户模式”页面修改用户密码。

3.2.2 插件信息

用户可在当前页面查看各个插件信息，包括作者、版本等，本节不进行详细描述。

3.2.3 末端插件

用户可在“工具配置 > 末端插件”安装夹爪插件，DobotSCStudio自带多种插件，插件路径为“安装路径/DobotSCStudio\endPackage”，用户可根据实际情况选择插件安装。

本节以DH插件为例进行说明。

步骤 1 在“末端插件”界面单击“安装”，勾选“DH-V2.zip”并单击“打开”安装DH夹爪。

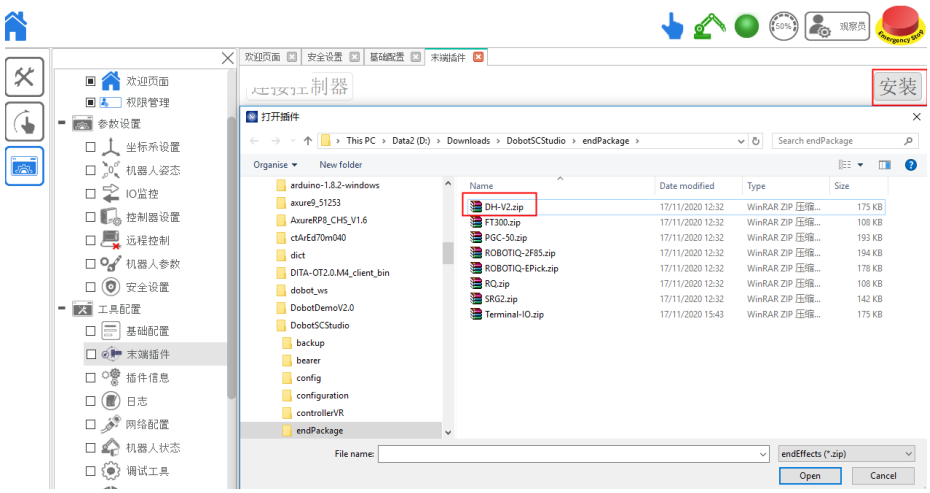


图 3.26 安装插件

安装完成后界面如图 3.27所示。

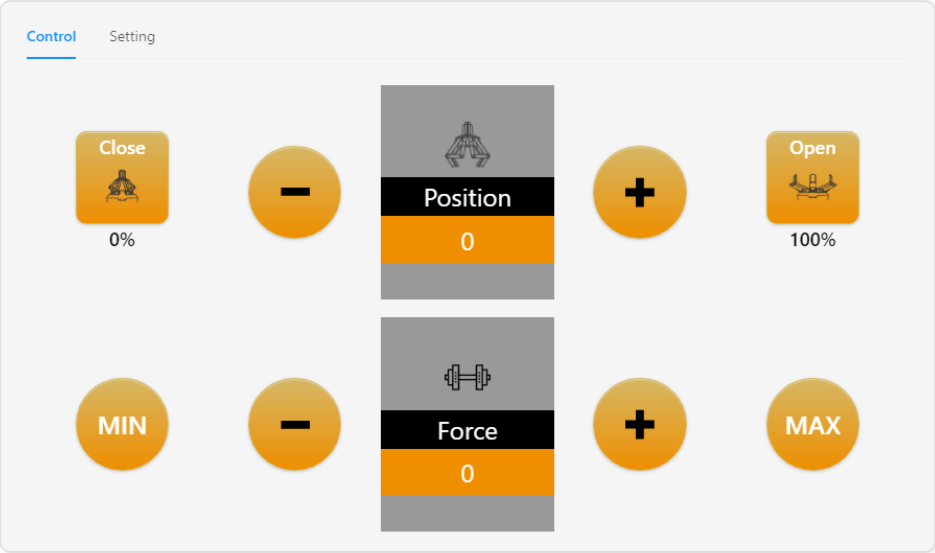


图 3.27 安装完成

步骤 2 在“Setting”界面设置“Baud”为115200，并单击Init。

步骤 3 在“Control”页面设置张合力度和张合位置。

3.2.4 日志

用户可以通过查看日志了解机械臂历史操作。可根据用户操作、控制器错误、伺服错误三类日志进行日志的筛选。点击“重置”可清除日志。

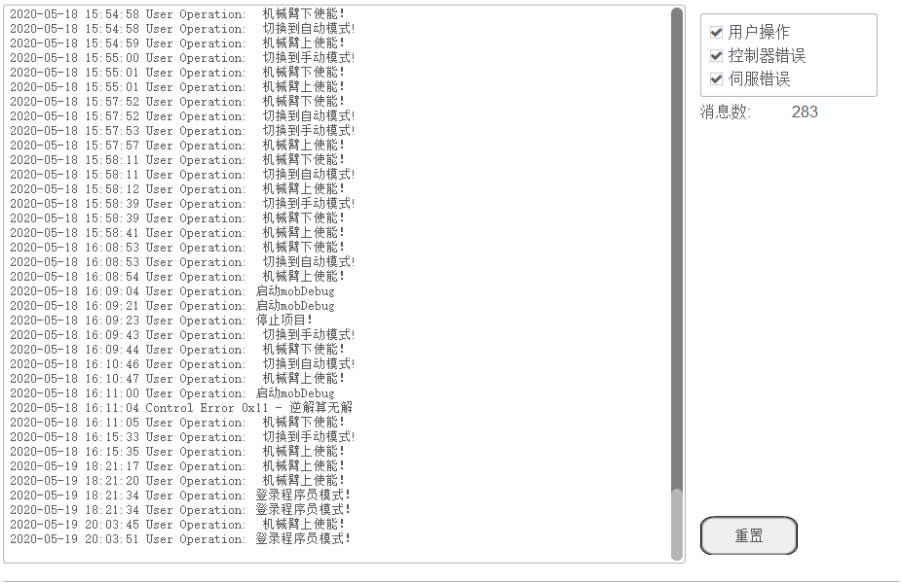


图 3.28 日志

3.2.5 网络配置

Dobot机器人控制系统可通过Ethernet接口与外部设备通信，支持TCP、UDP或Modbus协议。其默认IP地址为192.168.5.1。在实际应用中，如果采用TCP或UDP协议，则Dobot机器人

控制系统可作为客户端或服务端，如视觉场景；如果采用Modbus协议，则Dobot机器人控制系统作为Modbus从站，外部设备作为Modbus主站。

用户可在“ > 工具配置 > 网络配置”页面修改IP地址，如图 3.29所示。其中，Dobot机器人控制系统的IP地址必须与外部设备的IP地址在同一网段，且不冲突。

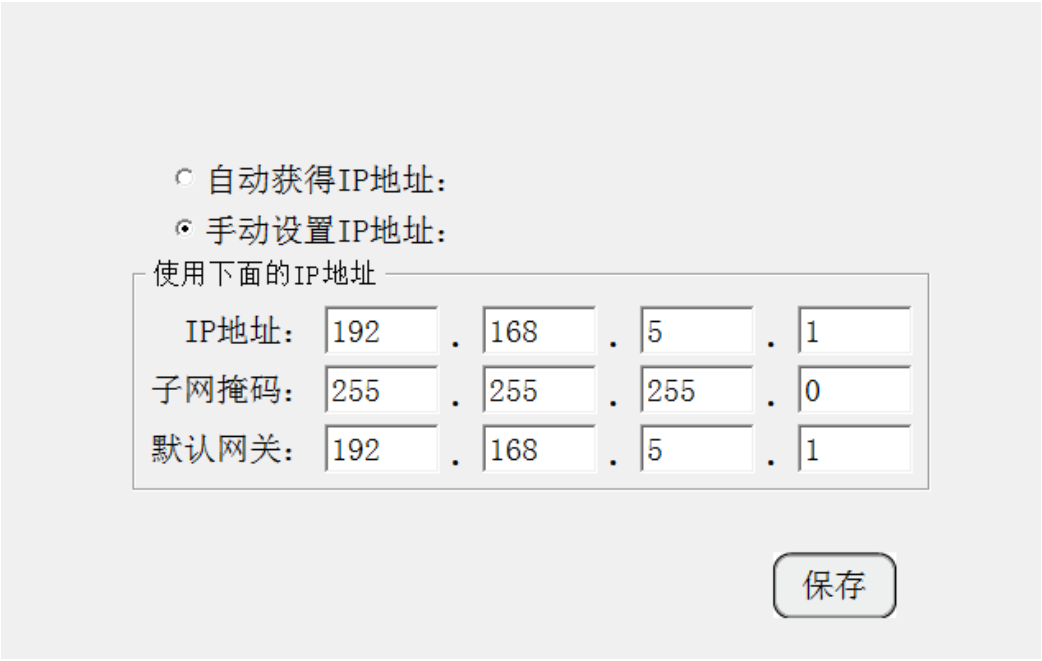


图 3.29 IP 地址设置

Dobot机器人控制系统与外部设备连接如图 3.30和图 3.31所示。

- 如果Dobot机器人控制系统与外部设备直接连接或通过交换机连接，则需勾选“手动设置IP地址”，修改IP地址、子网掩码以及默认网关后单击“保存”。
- 如果Dobot机器人控制系统与外部设备通过路由器，则勾选“自动获得IP地址”自动分配IP地址，然后单击“保存”。

⚠注意

通过路由器连接时请勿将网线接入“WAN”接口。

图 3.30 Dobot 机器人控制系统与外部设备直连

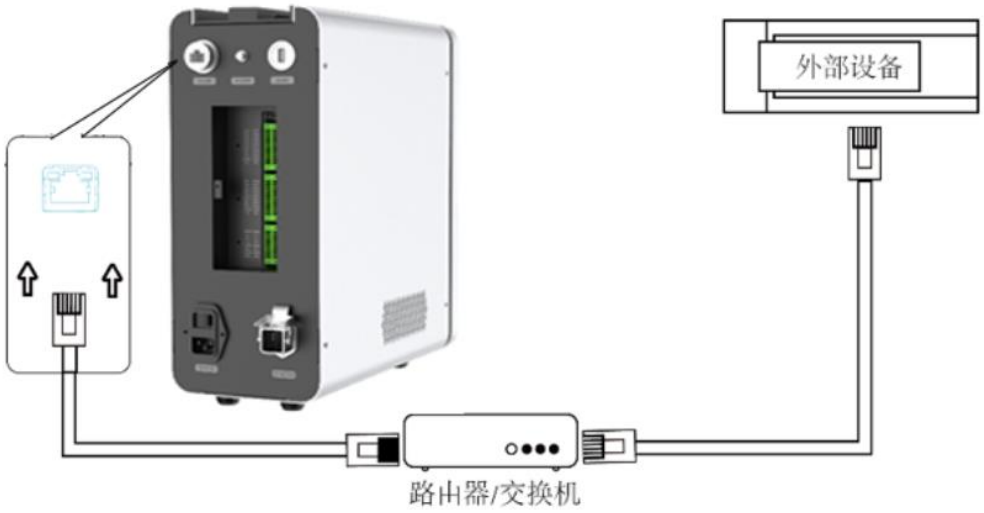


图 3.31 Dobot 机器人控制系统与外部设备通过路由器或交换机连接

3.2.6 机器人状态

DobotSCStudio支持查看机器人数据，包括控制器温度、电压、耗电量等，还可查看各关节负载，如图 3.32所示。

图 3.32 机器人状态

3.2.7 调试工具

DobotSCStudio提供串口调试、TCP调试、Modbus等多种调试工具，方便工作人员现场调试。调试方法本节不进行详细描述。

3.2.8 虚拟仿真

用户点动或运行机械臂时，可通过虚拟仿真界面实时查看机械臂运动状况，如图 3.33 所示。

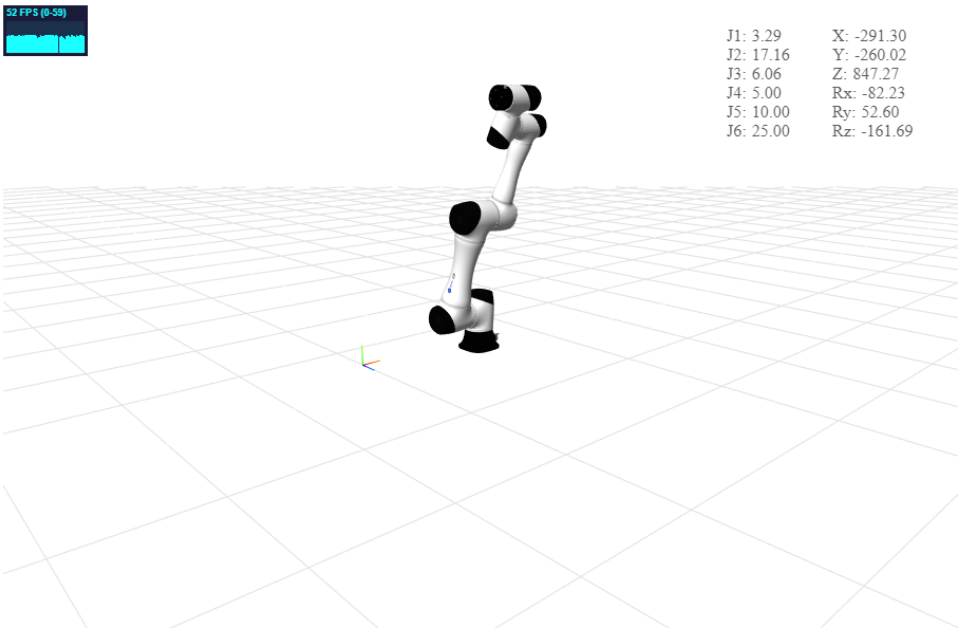


图 3.33 虚拟仿真

3.2.9 WiFi 设置

Dobot机器人控制系统可通过WiFi与外部设备通信。用户可在“ > 工具配置 >WiFi设置”页面修改WiFi名称和密码，重新启动控制柜后生效，如图 3.34所示。其中，WiFi初始密码为1234567890。

图 3.34 WiFi 设置

3.3 编程

3.3.1 工程说明

CR系列机械臂编程以工程形式来管理，包含了存点信息、全局变量、程序文件，工程结构如图 3.35所示。

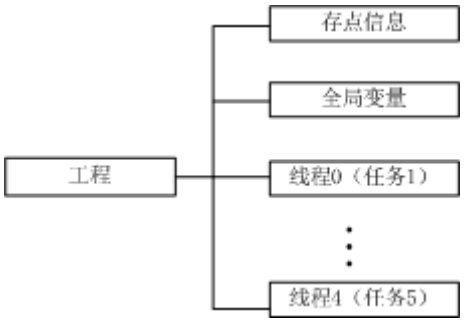


图 3.35 工程结构

3.3.2 编程面板说明

用户编写程序时，用户权限需为程序员以上模式，且需将控制柜切换至手动模式，其编程面板如图 3.36所示，详细说明如表 3.4所示。

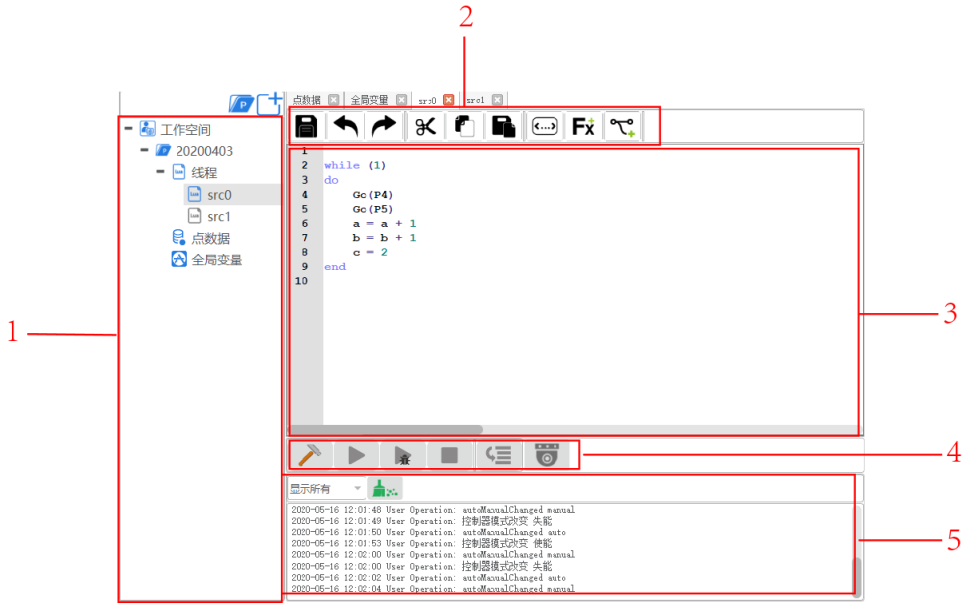


图 3.36 编程面板

表 3.4 编程面板说明

编号	说明
1	工程结构文件 • 点数据：示教存点，详细请参见*3.3.3.2 示教存点* • 全局变量：定义全局函数、全局点位、全局变量 • 线程：多任务代码文件。任务个数与新建工程时设置的“线程数”有关，最多可同时执行5

编号	说明
	个任务
2	编程常用按钮，详细说明请参见表 3.5
3	程序窗口
4	程序运行按钮，详细说明请参见表 3.7
5	调试结果显示

编程常用按钮说明如表 3.5所示。

表 3.5 编程常用按钮说明

图标	说明
	保存已编写的程序
	撤销编写的代码
	重做已编写的程序
	复制选中的代码
	剪切选中的代码
	粘贴已选中的代码
	常用API指令库，详细说明请参见4 编程语言
	用于代码注释
	常用运算指令和过程控制指令，详细说明请参见

3.3.3 编程说明

编程流程如图 3.37所示。

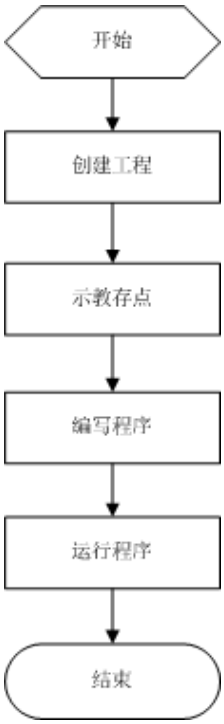


图 3.37 编程流程

3.3.3.1 创建工程

前提条件

- 机械臂已上电。
- 控制柜已切换至手动模式。

操作步骤

步骤 1 单击 进入编程页面，如图 3.38所示。

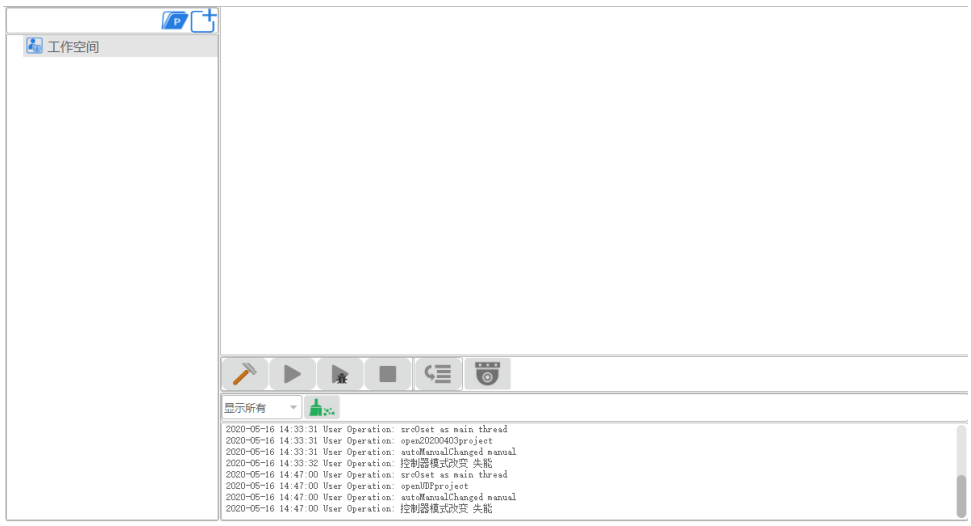


图 3.38 编程页面

步骤 2 单击。进入创建工程的页面，在创建工程页面输入工程名。也可选择工程模版。单击“确认”，编程页面会自动生成工程文件。

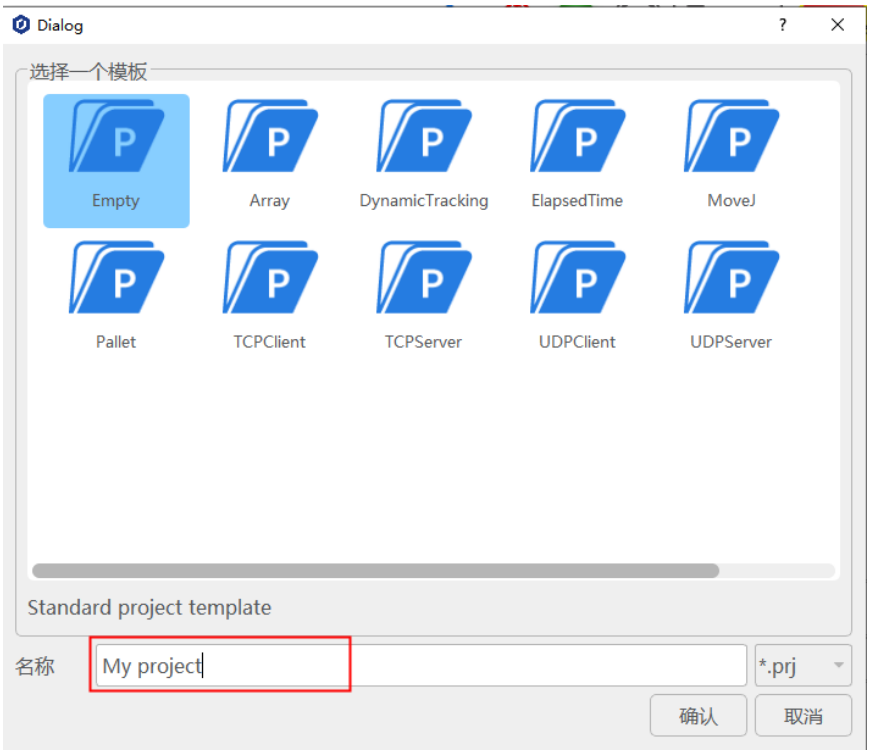


图 3.39 创建工程

步骤 3 根据实际需求创建多个任务，如图 3.40所示。单击“线程”右键选择“新建线程”，最多支持创建5个线程。

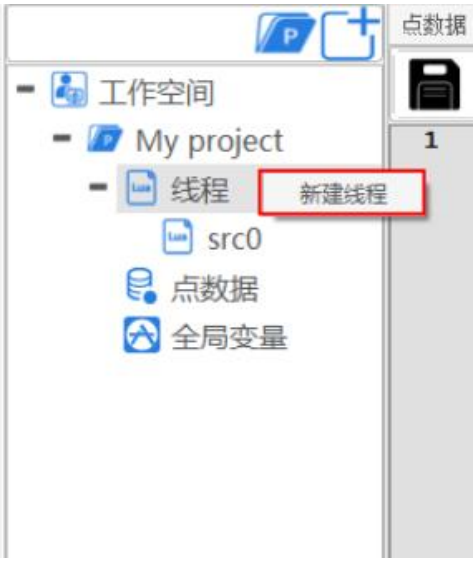


图 3.40 创建工程

步骤 4 （可选）导入已有工程的存点。

若用户需复用已有工程中的存点信息，可在创建工程页面右键单击“点数据”，单击“导入点数据”，然后选中工程后单击确认，如图 3.41所示。

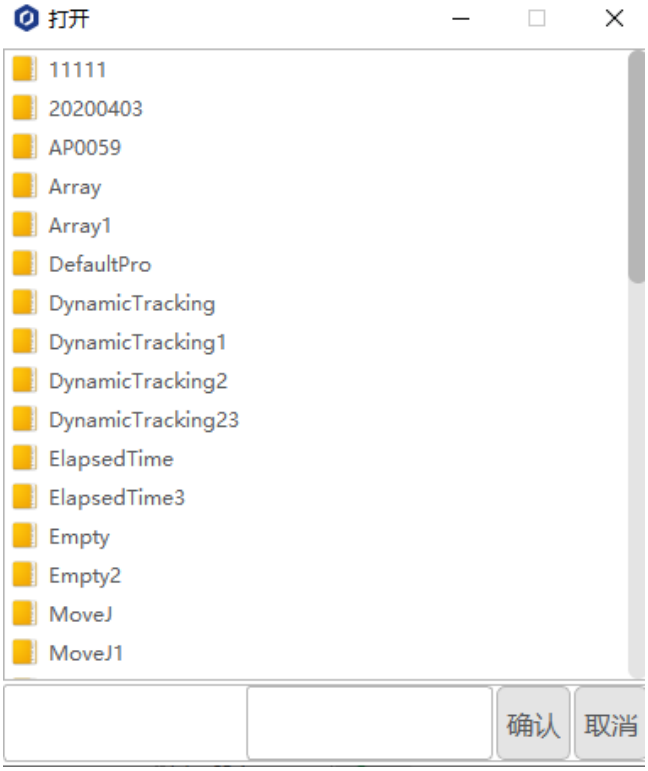


图 3.41 导入已有工程的存点信息

3.3.3.2 示教存点

前提条件

- 已创建工程或已导入工程。
- 控制柜已切换至手动模式。

操作步骤

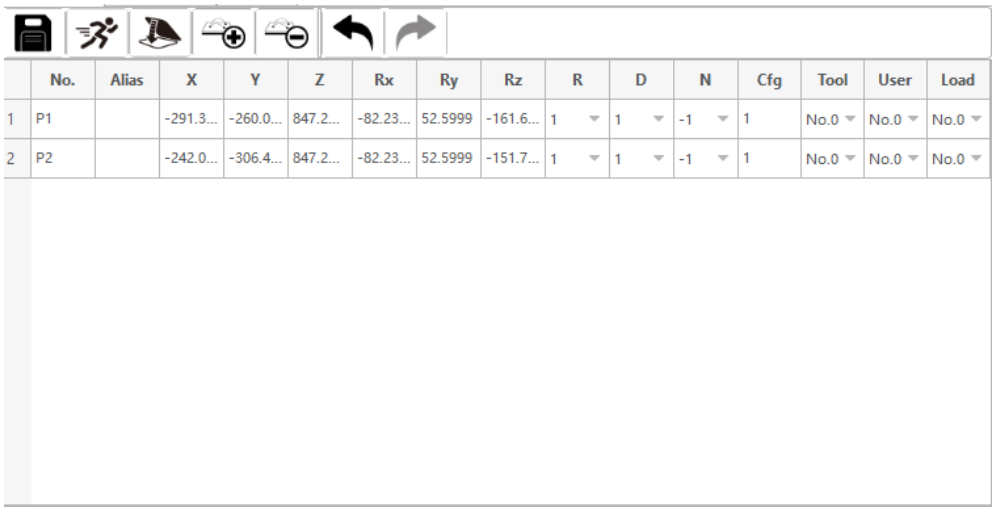
创建工程后需在“点数据”页面进行示教存点，以便在编写程序时调用。若在创建工程时已导入已有工程的存点信息，则可跳过该操作，编写程序时直接调用存点即可。

步骤 1 使能机械臂。

步骤 2 单击Jog按键，将机械臂移动至一点。

步骤 3 双击“点数据”，在“点数据”页面单击 新增存点信息。

“点数据”页面会显示存点信息，如图 3.42所示。



	No.	Alias	X	Y	Z	Rx	Ry	Rz	R	D	N	Cfg	Tool	User	Load
1	P1		-291.3...	-260.0...	847.2...	-82.23...	52.5999	-161.6...	1	1	-1	1	No.0	No.0	No.0
2	P2		-242.0...	-306.4...	847.2...	-82.23...	52.5999	-151.7...	1	1	-1	1	No.0	No.0	No.0

图 3.42 存点信息

表 3.6 示教按钮说明

按钮	说明
	添加示教点
	删除示教点
	覆盖点。选中示教点，点动机械臂至一点后，单击该图标可覆盖选中的示教点
	运行至示教点。勾选示教点，长按该按钮可使机械臂运行到勾选的示教点位置
	保存示教点
	撤销
	恢复

- 若用户需修改存点信息，则双击对应项修改即可。
- 若用户需覆盖当前点，则选中当前点后，单击 。

步骤 4 重复**步骤 2**和**步骤 3**新增存点。

步骤 5 单击，保存新增的存点。

3.3.3.3 编写程序

前提条件

- 已创建工程或导入工程。
- 已有存点信息。

操作步骤

CC系列控制柜采用Lua语言编程，封装了常用的运动指令以及基本的运算和逻辑指令，详细请参见4 编程语言。

假设已存示教点“P1”和“P2”。我们在线程“src0”页签编写一个简单的程序示例指导用户如何调用指令，其程序如图 3.43所示。该示例通过关节插补模式使机械臂在“P1”和“P2”点之间循环运动。

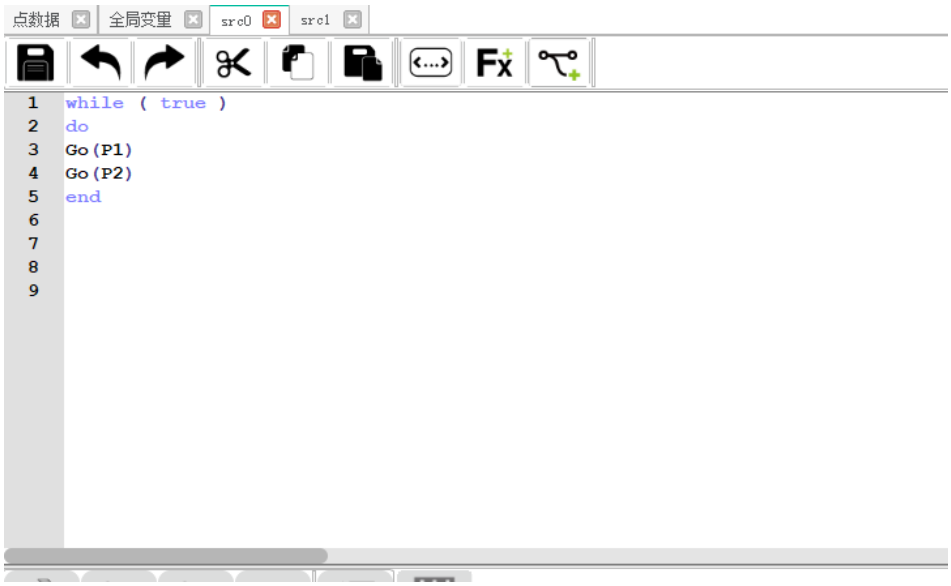


图 3.43 Lua 程序

步骤 1 单击“ >Syntax”，在语法界面双击击 while 调用循环指令，并将循环条件设置为“true”。

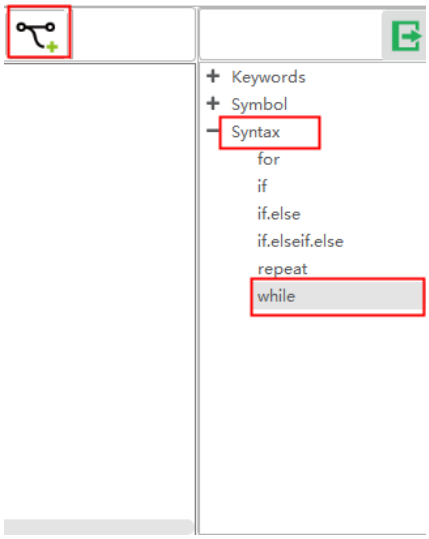


图 3.44 while 指令

步骤 2 在**while**循环**do…end**中添加运动指令。

1. 单击“Fx > MOVE”。

   显示运动指令列表，如图 3.45所示。

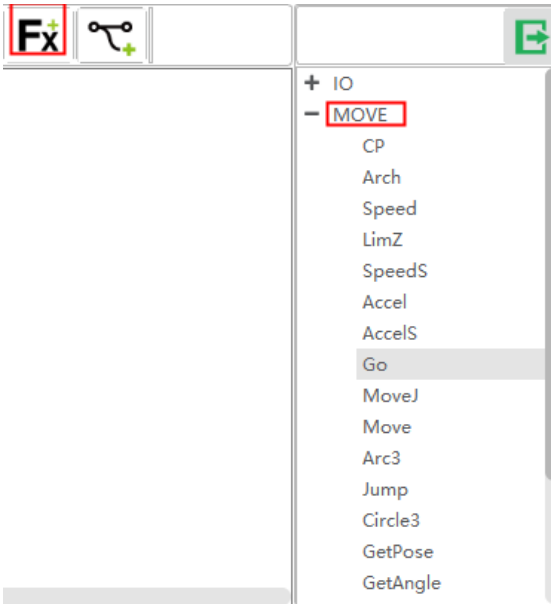


图 3.45 运动指令列表

2. 在“MOVE”中选择一条运动指令。

   此时会弹出该运动指令的参数设置页面。

   以**Go**指令为例，我们可以设置该指令所要运行的点。

3. 在**Go**指令的参数设置页面的“第一个参数”区域选择“P1”，单击“插入”。

表示通过Go模式运动至P1点。

图 3.46 调用 Go 指令

4. 换行，重复执行2。

5. 在**Go**指令的参数设置页面的“第一个参数”区域选择“P2”，单击“插入”。表示通过Go模式运动至P2点。

说明

如果用户需通过断点调试程序，则需在编写程序时设置断点。单击对应行编号附近的位置即可设置断点。

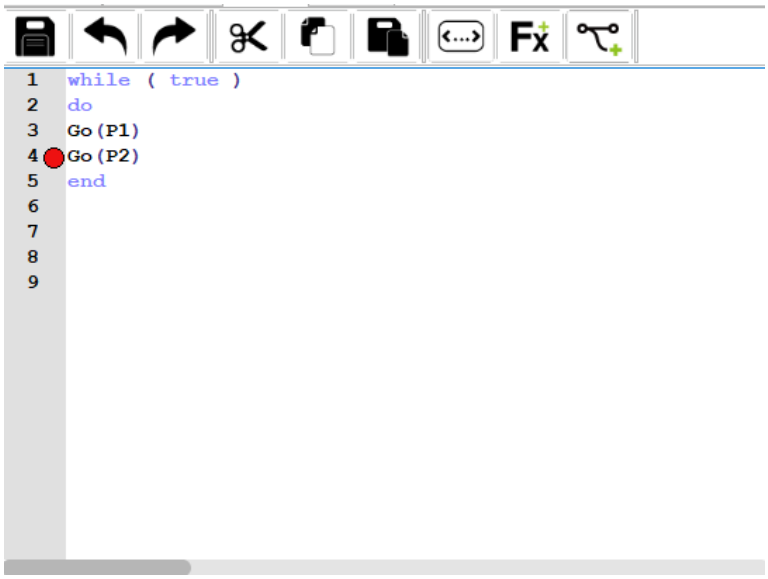


图 3.47 设置断点

步骤 3 单击。

此时，一个简单的程序已经编写完成。

3.3.3.4 运行程序

步骤 1 将机械臂切换至自动模式。

步骤 2 单击，使能机械臂。

此时，编程界面如图 3.48所示。

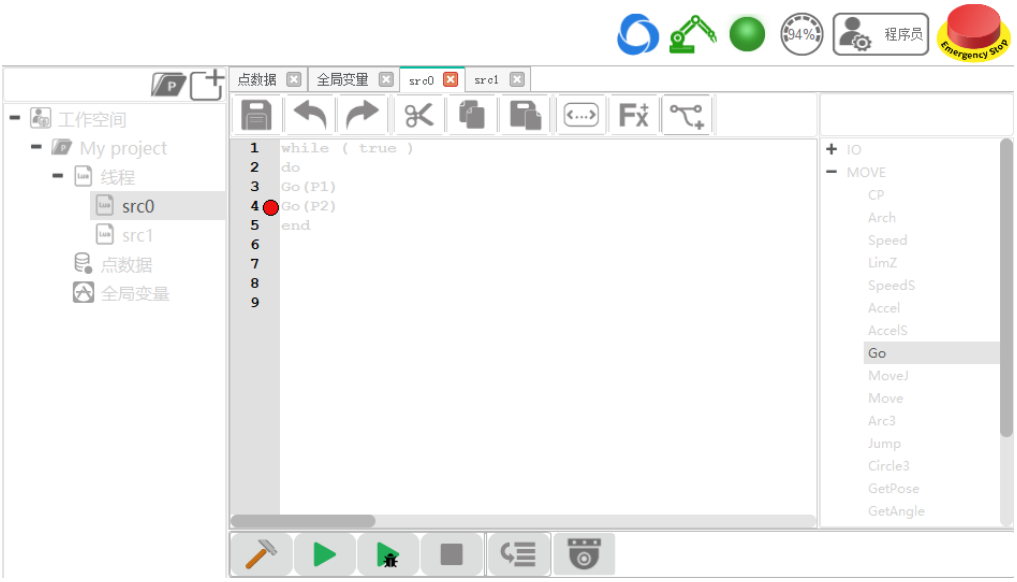


图 3.48 编程界面

其中，图 3.48中所示的编程运行按钮说明如表 3.7所示。

表 3.7 编程运行按钮说明

图标	说明
	构建程序 可检测代码是否正确
	一键运行程序 单击后， 变为 ，程序开始运行 如需暂停运行，则单击 即可
	开始运行程序 单击一次，表示启动调试， 变为 单击两次，表示开始运行程序， 变为 如需暂停运行，则单击 即可
	停止程序运行

图标	说明
	单步执行程序 仅当 变为 ，才能开始单步执行
	监控 调试程序时可实时监控调试过程

步骤 3 单击 即可运行程序，机械臂会根据编写的程序运行。

- 如果在编写程序时设置了断点，则程序运行至断点处的前一个位置后停止。如果需继续往下执行，则单击 后再单击 即可。
- 如果需单步执行，则单击 一次后，待 变为 ，即可单击 进行单步执行。

3.4 使能

- 手动模式下电机使能：手动模式下单击示教界面的使能图标 ，若 变为 ，此时可点动机械臂或对机械臂进行回零操作。
- 自动模式下电机使能：自动模式下单击示教界面的使能图标 ，若 变为 ，此时可通过运行程序控制机械臂运动。

3.5 设置全局速率

如果用户需全局修改机械臂点动和再现的运动速率，可在DobotSCStudio界面单击 或在“默认 > 整体速率”页面设置全局速度比率，如图 3.49所示。

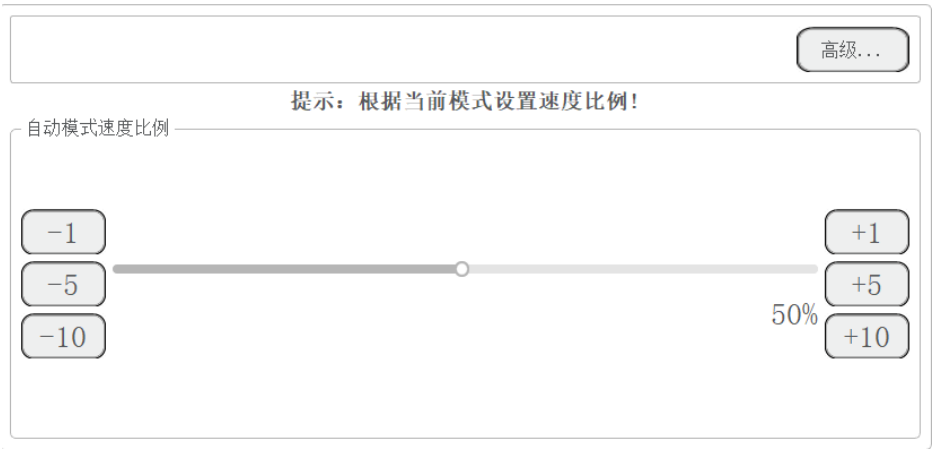


图 3.49 修改全局速率

点动或再现时，各轴（关节坐标系和笛卡尔坐标轴）速度和加速度计算方法如下所示：

- 各轴点动速度=各轴设置的点动最大速度*全局速率
- 各轴点动加速度=各轴设置的点动最大加速度*全局速率
- 各轴再现速度=各轴设置的再现最大速度*全局速率*指令中设置的百分比
- 各轴再现加速度=各轴设置的再现最大加速度*全局速率*指令中设置的百分比
- 各轴再现加加速度=各轴设置的再现最大加加速度*全局速率*指令中设置的百分比

说明

- 各轴点动或再现的最大速度、加速度或加加速度可在“参数设置”界面进行设置，详细请参见*3.1.7 机器人参数设置*。
- 指令设置的百分比可在编程时调用速度函数指令设置。

3.6 报警说明

当点动或存点的方法不正确或机械臂使用不当时，比如使机械臂限位或处于奇异点，会触发报警。

如果机械臂运行过程中触发报警，DobotSCStudio界面的报警图标 会变为 ，此时可在报警界面查看报警信息，如图 3.50所示。

清除报警方法说明：

- 如果是限位报警，则将控制柜切换至手动模式并使能，在关节坐标系下反向点动限位的轴即可清除报警。
- 如果是其他报警，则将控制柜切换至手动模式，然后在报警界面单击 清除报警说明。若无法清除报警，则重新启动机械臂。

图 3.50 报警界面

3.7 末端负载

为充分发挥机械臂自身具备的性能，需将末端负载重量与惯量设定在允许的范围内。通过合理的设定，可优化机械臂的运动，抑制振动，缩短作业时间。

末端负载重量即夹具末端重量与工件重量之和，用户可在单击使能图标时弹出的窗口中进行设置，如图 3.51所示。

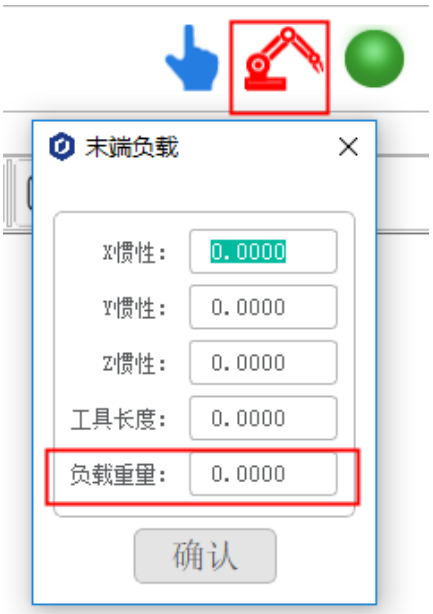


图 3.51 末端负载

4. 编程语言

Lua语言是CC系列控制柜采用的编程语言，用户可根据实际工艺需求，使用Lua语言在APP器上灵活的编写相应的控制程序。CC系列控制柜还封装了Dobot API指令供用户直接调用，简化了编程。本节描述常用的编程指令供用户参考。

4.1 算术运算符

表 4.1 算术运算符

指令符号	说明
+	加法运算
-	减法运算
*	乘法运算
/	浮点除法运算
//	向下取整除法运算
%	取余除法运算
^	指数运算
&	与运算
|	或运算
~	异或运算
<<	左移运算
>>	右移运算

4.2 关系运算符

表 4.2 关系运算符

指令符号	说明
==	等于
~=	不等于
<=	小于等于
>=	大于等于
<	小于
>	大于

4.3 逻辑运算符

表 4.3 逻辑运算符

指令符号	说明
or	逻辑或
not	逻辑非
and	逻辑与

4.4 一般关键字

表 4.4 一般关键字

指令符号	说明
break	跳出循环
local	定义局部变量，仅在定义的脚本中有效
nil	变量为空值
return	函数调用返回
enter	换行

4.5 一般符号

表 4.5 一般符号

指令符号	说明
#	求table数组长度

4.6 过程控制指令

表 4.6 过程控制指令

指令符号	说明
if…then…elseif…then…else…end	if条件分支指令
while…do…end	while循环控制指令
for…do…end	for循环控制指令
repeat… until()	repeat循环控制指令

4.7 全局变量

机械臂全局变量可在“global.lua”文件中定义，包括定义全局函数、全局点位、全局变量等。

- 全局函数：

```
function exam()
    print("This is an example")
end
```

- 以六轴为例，定义关节坐标点位，臂方向R=1，D=-1，N=0，第六轴角度标识Cfg=1，工具坐标系和用户坐标系都为0。

```
P = {armOrientation = {1, 1, -1, 1}, joint = {20,10,22,2.14,0.87,3.85}, tool = 0, user = 0}
```

- 全局变量

```
flag = 0
```

4.8 运动指令

表 4.7 运动指令

指令符号	说明
Go	从当前位置以点到点方式运动至笛卡尔坐标系下的目标位置
MoveJ	从当前位置以点到点方式运动至目标关节角度
Move	从当前位置以直线方式运动至笛卡尔坐标系下的目标位置
Arc3	从当前位置以圆弧插补方式移动至笛卡尔坐标系下的目标位置
Jump	从当前位置以Move运动方式进行门型运动，使机械臂移动至目标位置
Circle3	从当前位置以整圆插补方式移动至笛卡尔坐标系下的目标位置
RP	笛卡尔坐标系下增加X、Y、Z方向上的偏移量并返回一个新的坐标点
RJ	关节坐标系下增加每个轴的偏移量，并返回一个新的坐标点
MoveR	从当前位置以直线方式运动至笛卡尔坐标系下的偏移位置
GoR	从当前位置以点到点方式运动至笛卡尔坐标系下的偏移位置
MoveJR	从当前位置以点到点方式运动至关节偏移位置

⚠注意

所有指令中的可选参数在编程时可根据实际情况进行设置。如果设置，则只对当前指令有效。

表 4.8 Go 指令

原型	Go(P,” User=*1* Tool=*2* CP=*1* Speed=*50* Accel=*20* *SYNC*=1 ”)
描述	从当前位置以点到点方式运动至笛卡尔坐标系下的目标位置
参数	必选参数：P，表示目标点，可从“点数据”页面获取，也可自定义点位，但只支持笛卡尔坐标点位 可选参数： • User：表示用户坐标系，取值范围：0~9 • Tool：表示工具坐标系，取值范围：0~9 • CP：运动时设置平滑过渡，取值范围：0~100 • Speed：运动速度比例，取值范围：1~100 • Accel：运动加速度比例，取值范围：1~100 • SYNC：同步标识，取值范围：0或1。SYNC = 0表示异步执行，调用后立即返回，但不关注指令执行情况；SYNC = 1表示同步执行，调用后，待指令执行完才返回
示例	机械臂以默认设置运动至P1点 Go(P1)

表 4.9 MoveJ 指令

原型	MoveJ(P,”CP=*1* Speed=*50* Accel=*20* *SYNC*=1”)
描述	从当前位置以点到点方式运动至目标关节角度
参数	必选参数：P，表示目标关节角度，不能通过“点数据”页面导入，需先定义关节点位 可选参数： • CP：运动时设置平滑过渡，取值范围：0~100 • Speed：运动速度比例，取值范围：1~100 • Accel：运动加速度比例，取值范围：1~100 • SYNC：同步标识，取值范围：0或1。SYNC = 0表示异步执行，调用后立即返回，但不关注指令执行情况；SYNC = 1表示同步执行，调用后，待指令执行完才返回
示例	local P = {joint={0,-0.0674194,0,0}} MoveJ(P) 自定义关节坐标点点位P，机械臂设置运动至P点

表 4.10 Move 指令

原型	Move(P,” User=*1* Tool=*2* CP=*1* SpeedS=*50* AccelS=*20* *SYNC*=1”)
描述	从当前位置以直线方式运动至笛卡尔坐标系下的目标位置

参数	必选参数：P，表示目标点，可从“点数据”页面获取，也可自定义点位，但只支持笛卡尔坐标点位 可选参数： • User：表示用户坐标系，取值范围：0~9 • Tool：表示工具坐标系，取值范围：0~9 • CP：运动时设置平滑过渡，取值范围：0~100 • SpeedS：运动速度比例，取值范围：1~100 • AccelS：运动加速度比例，取值范围：1~100 • SYNC：同步标识，取值范围：0或1。SYNC = 0表示异步执行，调用后立即返回，但不关注指令执行情况；SYNC = 1表示同步执行，调用后，待指令执行完才返回
示例	机械臂以默认设置运动至P1点 Move(P1)

表 4.11 Arc3 指令

原型	Arc3(P1,P2, ” User=*1* Tool=*2* CP=*1* SpeedS=*50* AccelS=*20* *SYNC*=1”)
描述	从当前位置以圆弧插补方式移动至笛卡尔坐标系下的目标位置 该指令需结合其他运动指令确定圆弧起始点
参数	必选参数： • P1，表示圆弧中间点，可从“点数据”页面获取，也可自定义点位，但只支持笛卡尔坐标点位 • P2：表示圆弧结束点，可从“点数据”页面获取，也可自定义点位，但只支持笛卡尔坐标点位 可选参数： • User：表示用户坐标系，取值范围：0~9 • Tool：表示工具坐标系，取值范围：0~9 • CP：运动时设置平滑过渡，取值范围：0~100 • SpeedS：运动速度比例，取值范围：1~100 • AccelS：运动加速度比例，取值范围：1~100 • SYNC：同步标识，取值范围：0或1。SYNC = 0表示异步执行，调用后立即返回，但不关注指令执行情况；SYNC = 1表示同步执行，调用后，待指令执行完才返回

示例	While true do
	  Go(P1)
	  Arc3(P2,P3)
	end
	机械臂循环从P1点以圆弧的方式经过P2点运动至P3点

表 4.12 Jump 指令

原型	Jump(P," User=*1* Tool=*2* SpeedS=*50* AccelS=*20* Start=*10* Zlimit=*80* End=*50* *SYNC*=1")
描述	从当前位置以Move运动方式进行门型运动，使机械臂移动至目标位置
参数	必选参数：P，表示目标点，可从“点数据”页面获取，也可自定义点位，但只支持笛卡尔坐标点位。且P点高度不能高于Zlimit，否则会触发JUMP参数错误报警 可选参数： • User：表示用户坐标系，取值范围：0~9 • Tool：表示工具坐标系，取值范围：0~9 • SpeedS：运动速度比例，取值范围：1~100 • AccelS：运动加速度比例，取值范围：1~100 • Arch：设置Jump门型参数编号，取值范围：0~9 • Start：起始点高度 • Zlimit：最大抬升高度 • End：结束点下降高度 • SYNC：同步标识，取值范围：0或1。SYNC = 0表示异步执行，调用后立即返回，但不关注指令执行情况；SYNC = 1表示同步执行，调用后，待指令执行完才返回
示例	Jump(P1)：机械臂以Jump模式运动至P1点

表 4.13 Circle3 指令

原型	Circle3(P1,P2, Count, "User=*1* Tool=*2* CP=*1* SpeedS=*50* AccelS=*20* *SYNC*=1")
描述	从当前位置以整圆插补方式移动至笛卡尔坐标系下的目标位置 该指令需结合其他运动指令确定圆弧起始点
参数	必选参数： • P1，表示整圆中间点，可从“点数据”页面获取，也可自定义点位，但只支持笛卡尔坐标点位 • P2：表示整圆结束点，可从“点数据”页面获取，也可自定义点位，但只支持笛卡尔坐标点位 • Count：整圆个数，取值范围：1~999 可选参数： • User：表示用户坐标系，取值范围：0~9 • Tool：表示工具坐标系，取值范围：0~9 • CP：运动时设置平滑过渡，取值范围：0~100 • SpeedS：运动速度比例，取值范围：1~100 • AccelS：运动加速度比例，取值范围：1~100 • SYNC：同步标识，取值范围：0或1。SYNC = 0表示异步执行，调用后立即返回，但

	不关注指令执行情况；SYNC = 1表示同步执行，调用后，待指令执行完才返回
示例	Go(P1) Circle3(P2,P3,1) 机械臂从P1点以整圆的方式经过P2点运动至P3点

表 4.14 RP 指令

原型	RP(P1, {OffsetX, OffsetY, OffsetZ})
描述	笛卡尔坐标系下增加X、Y、Z方向上的偏移量并返回一个新的笛卡尔坐标点 除MoveJ外，其他运动指令均支持运行至该点
参数	• P1，表示当前笛卡尔坐标点，可从“点数据”页面获取，也可自定义点位，但只支持笛卡尔坐标点位 • OffsetX，OffsetY，OffsetZ：笛卡尔坐标系下X轴、Y轴、Z轴方向上的偏移 单位：毫米
返回	笛卡尔坐标点
示例	P2=RP(P1, {50,10,32}) Move(P2)或 Move(RP(P1, {50,10,32}))

表 4.15 RP 指令

原型	RJ(P1, {Offset1, Offset2, Offset3, Offset4, Offset5, Offset6})
描述	关节坐标系下增加每个轴的偏移量，并返回一个新的关节坐标点 仅支持MoveJ运动指令运行至该点
参数	• P1，表示当前关节点位，不能通过“点数据”页面导入，需先定义 • Offset1~Offset6：关节坐标系下J1轴 ~ J6轴方向上的偏移。 单位：度
返回	关节坐标点
示例	local P1 = {joint={0,-0.0674194,0,0}} P2=RJ(P1, {60,50,32,30}) MoveJ(P2)或 MoveJ(RJ(P1, {60,50,32,30}))

表 4.16 GoR 指令

原型	GoR({OffsetX, OffsetY, OffsetZ},” User=*1* Tool=*2* CP=*1* Speed=*50* Accel=*20* *SYNC*=1 ”)
描述	从当前位置以点到点方式运动至笛卡尔坐标系下的偏移位置
参数	必选参数：OffsetX， OffsetY， OffsetZ：笛卡尔坐标系下X轴、Y轴、Z轴方向上的偏移

	单位：毫米 可选参数： • User：表示用户坐标系，取值范围：0~9 • Tool：表示工具坐标系，取值范围：0~9 • Speed：运动速度比例，取值范围：1~100 • Accel：运动加速度比例，取值范围：1~100 • SYNC：同步标识，取值范围：0或1。SYNC = 0表示异步执行，调用后立即返回，但不关注指令执行情况；SYNC = 1表示同步执行，调用后，待指令执行完才返回
示例	Go(P1) GoR({10,10,10},"Accel=100 Speed=100 CP=100")

表 4.17 MoveJR 指令

原型	MoveJR({Offset1, Offset2, Offset3, Offset4, Offset5, Offset6},” CP=*1* Speed=*50* Accel=*20* *SYNC*=1”)
描述	从当前位置以点到点方式运动至关节偏移角度
参数	必选参数：Offset1~Offset6：关节坐标系下J1轴 ~ J6轴方向上的偏移 单位：度 可选参数： • CP：运动时设置平滑过渡，取值范围：0~100 • Speed：运动速度比例，取值范围：1~100 • Accel：运动加速度比例，取值范围：1~100 • SYNC：同步标识，取值范围：0或1。SYNC = 0表示异步执行，调用后立即返回，但不关注指令执行情况；SYNC = 1表示同步执行，调用后，待指令执行完才返回
示例	Go(P1) MoveJR({20,20,10,0},"SYNC=1")

表 4.18 Move 指令

原型	MoveR({OffsetX, OffsetY, OffsetZ},” User=*1* Tool=*2* CP=*1* SpeedS=*50* AccelS=*20* *SYNC*=1”)
描述	从当前位置以直线方式运动至笛卡尔坐标系下偏移位置
参数	必选参数：必选参数：OffsetX, OffsetY, OffsetZ： 笛卡尔坐标系下X轴、Y轴、Z轴方向上的偏移 单位：毫米 可选参数： • User：表示用户坐标系，取值范围：0~9

	• Tool：表示工具坐标系，取值范围：0~9 • SpeedS：运动速度比例，取值范围：1~100 • AccelS：运动加速度比例，取值范围：1~100 • SYNC：同步标识，取值范围：0或1。SYNC = 0表示异步执行，调用后立即返回，但不关注指令执行情况；SYNC = 1表示同步执行，调用后，待指令执行完才返回
示例	Go(P1) MoveR({20,20,20},"AccelS=100 SpeedS=100 CP=100")

4.9 运动参数设置指令

表 4.19 运动参数设置指令

指令符号	说明
Accel	设置Go、MoveJ、GoR、MoveJR运动加速度比例
AccelS	设置Move、Jump、Arc3、Circle3、MoveR运动加速度比例
Speed	设置Go、MoveJ、GoR、MoveJR运动速度比例
SpeedS	设置Move、Jump、Arc3、Circle3、MoveR运动速度比例
Arch	设置Jump门型参数索引（起始点抬升高度、最大抬升高度、结束点下降高度）
CP	运动时设置平滑过渡
LimZ	设置Jump模式最大抬升高度

表 4.20 Accel 指令

原型	Accel(R)
描述	设置Go、Jump、MoveJ、GoR、MoveJR运动加速度比例
参数	R：百分比，取值范围1~100
示例	Accel(50) Go(P1) 表示机械臂以50%的加速度比例运动至P1点

表 4.21 AccelS 指令

原型	AccelS(R)
描述	设置Move、Arc3、Circle3、MoveR运动加速度比例

原型	AccelS(R)
参数	R：百分比，取值范围1~100
示例	AccelS(20) Move(P1) 表示机械臂以20%的加速度比例运动至P1点

表 4.22 Speed 指令

原型	Speed(R)
描述	设置Go、Jump、MoveJ、GoR、MoveJR运动速度比例
参数	R：百分比，取值范围1~100
示例	Speed(20) Go(P1) 表示机械臂以20%的速度比例运动至P1点

表 4.23 SpeedS 指令

原型	SpeedS(R)
描述	设置Move、Arc3、Circle3、MoveR运动速度比例
参数	R：百分比，取值范围1~100
示例	SpeedS(20) Move(P1) 表示机械臂以20%的速度比例运动至P1点

表 4.24 Arch 指令

原型	Arch(Index)
描述	设置Jump门型参数索引（起始点抬升高度、最大抬升高度、结束点下降高度） Jump门型参数需在“参数设置 > 机器人参数设置 > 门型参数”中设置，详细请参见。
参数	Index：门型参数索引，取值范围：0~9 需在“参数设置 > 机器人参数设置 > 门型参数”勾选对应索引后才能生效
示例	Arch(1) Jump(P1)

表 4.25 CP 指令

原型	CP(R)

描述	设置平滑过渡比例，即从起始点经过中间点到达终点时，经过中间点是以直角方式过渡还是以曲线方式过渡，如图 4.1所示 仅对Go、Move、Arc3、Circle3、GoR、MoveR有效
参数	R：平滑过渡比例。取值范围：0~100 “0”表示运动时不开启平滑过渡
示例	CP(50) Move(P1) Move(P2) Move(P3) 机械臂从P1运动至P3点经过P2点时以50%平滑过渡

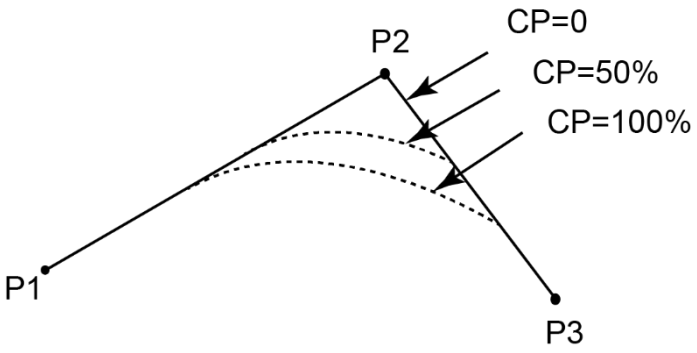


图 4.1 CP 平滑度

表 4.26 LimZ 指令

原型	LimZ(zValue)
描述	设置Jump模式最大抬升高度
参数	zValue：最大抬升高度，不能超过机械臂Z轴极限位置
示例	LimZ(80) Jump(P,” Start=*10* Zlimit=LimZ End=*50”*)

4.10 六维传感器运动指令

表 4.27 六维传感器指令

指令符号	说明
SixForceHome	六维传感器回零指令
Spiral	六维传感器螺旋运动指令
Rotation	六维传感器旋转运动指令

指令符号	说明
Linear	六维传感器直线运动指令

表 4.28 回零六维传感器

原型	SixForceHome
描述	对六维传感器进行回零操作
参数	无
示例	SixForceHome()：运行该指令回零六维传感器

表 4.29 六维传感器螺旋运动

原型	Spiral(P, User, Tool, Direction, SpeedC, Force, Insertion, Perturn, PeckMode, MaxValue)
描述	机械臂在当前位置和指定位置之间做螺旋运动寻找孔位，该指定点要求离孔位距离交近，是孔位探索的起始点
参数	• P：指定的点位 • User：表示用户坐标系，取值范围：0~9 • Tool：表示工具坐标系，取值范围：0~9 • Direction: 选定的插孔方向(0：正向；1：反向) • SpeedC: 插孔的速度(mm/s) • Force: 螺旋阈值(N) • Insertion: 插孔阈值(N) • Perturn: 螺旋半径(mm) • PeckMode: 点接触模式(ON/OFF) • MaxValue: 最大螺旋半径(mm)
示例	Spiral(P1,User=1 Tool=2 Dirction=0 SpeedC=5 Force =10 Insertion=3 Perturn=0.7 PeckMode=OFF MaxValue =5”) 在当前位置和 P1 之间做螺旋运动寻找孔位。当插孔方向上的阻力大于 Force 阈值时，机器人做螺旋运动进行孔位探索，当插孔方向上的阻力小于 Insertion 阈值时，机器人沿着插孔方向运动，进行插孔工作。

表 4.30 六维传感器旋转运动

原型	Rotation (P, User, Tool, Direction, SpeedC, Force, RotationSpeed, MaxTorque, PeckMode, MaxValue)

描述	机械臂在当前位置和指定位置之间做旋转运动寻找孔位，该指定点要求离孔位距离交近，是孔位探索的起始点
参数	• P：指定的点位 • User：表示用户坐标系，取值范围：0~9 • Tool：表示工具坐标系，取值范围：0~9 • Direction: 选定的插孔方向(0：正向；1：反向) • SpeedC: 插孔的速度(mm/s) • Force: 旋转阈值(N) • RotationSpeed: 旋转速度(°/s) • MaxTorque: 最大力矩(Nm) • PeckMode: 点接触模式(ON/OFF) • MaxValue: 最大螺旋半径(mm)
示例	Rotation (P1, “User=1 Tool=2 Dirction=0 SpeedC =5 Force =10 RotationSpeed=5 MaxTorque=1 PeckMode=OFF MaxValue =45”) 在当前位置和 P1 之间做旋转运动进行插孔。当插孔方向上的阻力大于 Force 阈值时，机器人做旋转运动进行孔位探索，当插孔方向上的阻力小于 Force 阈值时，机器人沿着插孔方向运动，进行插孔工作。

表 4.31 六维传感器线性运动

原型	Linear (User, Tool, Direction, SpeedC, Force, MaxValue)
描述	机械臂在孔位方向做线性插孔运动
参数	• User：表示用户坐标系，取值范围：0~9 • Tool：表示工具坐标系，取值范围：0~9 • Direction: 选定的插孔方向(0：正向；1：反向) • SpeedC: 插孔的速度(mm/s) • Force: 螺旋阈值(N) • MaxValue: 最大螺旋半径(mm)
示例	Linear(“User=1 Tool=2 Dirction=0 SpeedC =5 Force=10 MaxValue=45”) 在当前孔位做线性插孔运动，当插入方向的阻力大于 Force 阈值，则认为插入完成。

4.11 输入/输出指令

表 4.32 输入/输出指令

指令符号	说明
DI	读取数字输入端口状态
DO	设置数字输出端口状态（队列指令）
DOExecute	设置数字输出端口状态（立即指令）

说明

Dobot机器人控制系统支持两种类型的指令：立即指令与队列指令。

- 立即指令：Dobot控制器在收到指令后立即处理该指令，而不管当前控制器是否在还在处理其他指令。
- 队列指令：Dobot控制器在收到指令后会将该指令放入控制器内部的指令队列中，Dobot控制器将顺序执行指令。

表 4.33 数字输入指令

原型	DI(*index*)
描述	读取输入端口状态
参数	index：数字输入索引，取值范围：1~32
返回	• 当设置了索引时，DI(index)返回对应的index的电平（ON/OFF） • 当没有参数时，返回所有index的电平并以table形式存储 假如local di=DI()，table存储形式为{num = 24 value = {0x55, 0xAA, 0x52}}，则可通过di.num，di.value[n]等来获取对应index的值
示例	if (DI(1))==ON then Move(P1) end 数字输入端口1为高电平时机械臂以直线运动方式运动至P1点

表 4.34 数字输出指令（立即指令）

原型	DOExecute(*index,ON | OFF*)
描述	设置数字输出端口状态（立即指令）
参数	• index：数字输出索引，取值范围：1~16 • ON/OFF：数字输出端口状态，ON：高电平；OFF：低电平
示例	DOExecute(1,OFF) 将数字输出端口1设置为低电平

表 4.35 数字输出指令（队列指令）

原型	DO(*index,ON* | *OFF*)
描述	设置数字输出端口状态（队列指令）
参数	• index：数字输出索引，取值范围：1~24 • ON/OFF：数字输出端口状态，ON：高电平；OFF：低电平
示例	DO(1,ON) 将数字输出端口1设置为高电平

4.12 程序管理指令

表 4.36 程序管理指令

指令符号	说明
Wait	运动暂停时间
Sleep	通用脚本暂停时间
Pause	程序运行暂停
ResetElapsedTime	开始计时
ElapsedTime	结束计时
System	获取当前时间

表 4.37 指令延时时间

原型	Wait(*time*)
描述	运动暂停时间
参数	time：延时时间。单位：毫秒
示例	Go(P1) Wait(1000) 机械臂执行至P1点后延时1000毫秒

表 4.38 程序等待时间

原型	Sleep(*time*)
描述	通用脚本暂停时间
参数	time：延时时间。单位：毫秒

示例	while true do Speed(100) Go(P1) sleep(3) Speed(100) Accel(40) Go(P2) sleep(3) end

表 4.39 程序运行暂停指令

原型	Pause()
描述	程序运行暂停指令 当程序运行至该指令后，程序暂停运行，此时APP程序运行页面中编程按钮 变为 。若需继续运行，则单击 即可
参数	无
示例	while true do Go(P1) Go(P2) Pause() Go(P3) Go(P4) end 机械臂执行至P2点后暂停运行

表 4.40 开始计时指令

原型	ResetElapsedTime()
描述	待此指令前所有指令执行完成后开始计时，需结合ElapsedTime()一起使用 如计算脚本中某段代码执行完成所耗时间
参数	无
返回	无

示例	Go(P2, " Speed=100 Accel=100") ResetElapsedTime() for i=1,10 do Jump(P1, " Speed=100 Accel=100 Start=0 End=0 ZLimit=185") Jump(P2, " Speed=100 Accel=100 Start=0 End=0 ZLimit=185") end print (ElapsedTime()) Sleep(1000)

表 4.41 结束计时指令

原型	ElapsedTime()
描述	结束计时，返回时间差。需结合ResetElapsedTime()一起使用
参数	无
返回	返回时间差，单位：毫秒
示例	Go(P2, " Speed=100 Accel=100") ResetElapsedTime() for i=1,10 do Jump(P1, " Speed=100 Accel=100 Start=0 End=0 ZLimit=185") Jump(P2, " Speed=100 Accel=100 Start=0 End=0 ZLimit=185") end print (ElapsedTime()) Sleep(1000)

表 4.42 获取当前时间

原型	Systime()
描述	获取当前时间
参数	无
返回	返回当前时间

示例	Go(P2, " Speed=100 Accel=100") local time1=Systime() for i=1,10 do Jump(P1, " Speed=100 Accel=100 Start=0 End=0 ZLimit=185") Jump(P2, " Speed=100 Accel=100 Start=0 End=0 ZLimit=185") end local time2=Systime() local time = time2 - time1 Sleep(1000)

4.13 获取位姿指令

表 4.43 位姿指令（1）

原型	GetPose()
描述	获取笛卡尔坐标系下机械臂的实时位姿 如果设置了用户坐标系或工具坐标系，则获取的位姿为当前坐标系下的位姿
参数	无
返回	机械臂当前位置的笛卡尔坐标值
示例	local currentPose = GetPose() --获取当前位置 local liftPose = {coordinate = {currentPose.coordinate[1], currentPose. coordinate[2], currentPose. coordinate[3],currentPose. coordinate[4] }, tool = currentPose.tool, user = currentPose.user} -- 原地抬升 Go(liftPose,"Speed=100 Accel=100") Go(P1)

表 4.44 位姿指令（2）

原型	GetAngle()
描述	获取关节坐标系下机械臂的实时位姿
参数	无
返回	机械臂当前位置的关节坐标值

示例	local armPose local joint = GetAngle() --获取当前位姿 local liftPose = { joint = {joint.joint[1], joint.joint[2], joint.joint[3], joint.joint[4]}, tool = 0, user = 0}

4.14 TCP

表 4.45 创建 TCP 网络指令

原型	err, socket = TCPCreate(*isServer*, *IP*, *port*)
描述	创建TCP网络 仅支持单连接
参数	isServer：是否创建服务器。0：表示创建客户端；1：表示创建服务端 IP：服务端IP地址，需与客户端IP地址在同一网段，且不冲突 port：服务端端口 机械臂作为服务端时，“port”不能设置为502或8080，否则会与Modbus默认端口或流水线跟踪中使用的端口冲突，导致创建TCP网络失败
返回	err： 0：创建TCP网络成功 1：创建TCP网络失败 socket：创建的socket对象
示例	请参见程序 4.1和程序 4.2

表 4.46 TCP 连接指令

原型	TCPStart(*socket*, *timeout*)
描述	TCP连接功能
参数	socket：socket对象 timeout：等待超时时间。单位：秒。如果为0，则一直等待连接。如果不为0，则超过设定的时间后退出连接
返回	• 0：TCP网络连接成功 • 1：输入参数错误 • 2：socket对象不存在 • 3：设置超时时间错误 • 4：若机械臂作为客户端，则说明连接错误；若机械臂作为服务端，则说明接收错误
示例	请参见程序 4.1和程序 4.2

表 4.47 TCP 接收数据指令

原型	err, Recbuf = TCPRead(*socket*, *timeout*, *type*)
描述	机械臂作为客户端接收来自服务端的数据 机械臂作为服务端时接收来自客户端的数据
参数	socket：socket对象 timeout：接收超时时间，单位：秒。若timeout为0或者不设置，则该指令为阻塞式读取，即接收完数据后程序才往下执行；若不为0，则超过设定的timeout后，程序继续往下执行，即不考虑数据是否接收完，继续往下执行 type：缓存类型。若设置为空，则RecBuf缓存格式为table形式，如果设置为“string”，则RecBuf缓存字符串形式
返回	err： 0：接收数据成功 1：接收数据失败 Recbuf：接收数据缓存区
示例	请参见程序 4.1和程序 4.2

表 4.48 TCP 发送数据指令

原型	TCPWrite(*socket*, *buf*, *timeout*)
描述	机械臂作为客户端时发送数据给服务端 机械臂作为服务端时发送数据给客户端
参数	socket：socket对象 buf：发送的数据 timeout：超时时间，单位：秒。若timeout为0或者不设置，则该指令为阻塞式读取，即发送完数据后程序才往下执行；若不为0，则超过设定的timeout后，程序继续往下执行，即不考虑数据是否接收完，继续往下执行
返回	0：发送数据成功 1：发送数据失败
示例	请参见程序 4.1和程序 4.2

表 4.49 关闭 TCP 指令

原型	TCPDestroy(*socket*)
描述	关闭TCP功能
参数	socket：socket对象

返回	0：关闭TCP成功 1：关闭TCP失败
示例	请参见程序 4.1和程序 4.2

⚠注意

当前TCP连接仅支持连接单个客户端。连接时请启动服务端后再连接客户端；断开连接时请先关闭客户端，以免服务端端口未及时释放导致再次连接时连接失败。

程序 4.1 TCP 服务端示例

```
local ip="192.168.5.1"                          //机械臂的 IP 地址作为服务端的 IP 地址
local port=6001                                 //服务端端口
local err=0
local socket=0
err, socket = TCPCreate(true, ip, port)
if err == 0 then
    err = TCPStart(socket, 0)
    if err == 0 then
        local RecBuf
        while true do
            TCPWrite(socket, "tcp server test")          //服务端发送数据至客户端
            err, RecBuf = TCPRead(socket,0,"string")     //服务端接收客户端发送的数据
            if err == 0 then
                Go(P1)                                   //服务端接收到数据后开始执行运动指令
                Go(P2)
                print(buf)
            else
                print("Read error ".. err)
                break
            end
        end
    else
        print("Create failed ".. err)
    end
    TCPDestroy(socket)
else
    print("Create failed ".. err)
end
```

程序 4.2 TCP 客户端示例

```
local ip="192.168.5.25"                                  //外部设备如相机的 IP 地址作为服务端的 IP 地址
local port=6001                                          //服务端端口
local err=0
local socket=0
err, socket = TCPCreate(false, ip, port)
if err == 0 then
    err = TCPStart(socket, 0)
    if err == 0 then
        local RecBuf
        while true do
            TCPWrite(socket, "tcp client test")                    //客户端发送数据至服务端
            TCPWrite(socket, {0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07})
            err, RecBuf = TCPRead(socket, 0)                       //客户端接收服务端发送的数据
            if err == 0 then
                Go(P1)                                   //客户端接收到数据后开始执行运动指令
                Go(P2)
                print(buf)
            else
                print("Read error ".. err)
                break
            end
        end
    else
        print("Create failed ".. err)
    end
    TCPDestroy(socket)
else
    print("Create failed ".. err)
end
```

4.50 读取单个点数据

原型	err, Recbuf = TCPReadVision(*socket, timeout，type)*
描述	读取单个点数据

参数	socket：socket对象 timeout：接收超时时间，单位：秒。若timeout为0或者不设置，则该指令为阻塞式读取；若不为0，则超过设定的timeout后，程序继续往下执行 type：缓存类型。若设置为空，则RecBuf缓存格式为table形式，如果设置为“string”，则RecBuf缓存字符串形式
返回	err： 0：接收数据成功 1：接收数据失败 Recbuf：接收数据缓存区
示例	local err, RecBuf = TCPReadVision (socket, timeout, “string”); 其中，RecBuf内容为{ coordinate={x，y，z，r}}

4.51 读取多个点数据

原型	err, Recbuf = TCPReadMultiVision(*socket, timeout，type*)
描述	读取多个点数据
参数	socket：socket对象 timeout：接收超时时间，单位：秒。若timeout为0或者不设置，则该指令为阻塞式读取；若不为0，则超过设定的timeout后，程序继续往下执行 type：缓存类型。若设置为空，则RecBuf缓存格式为table形式，如果设置为“string”，则RecBuf缓存字符串形式
返回	err： 0：接收数据成功 1：接收数据失败 Recbuf：接收数据缓存区
示例	local err, RecBuf = TCPReadMultiVision (socket, timeout, “string”); 其中，RecBuf内容为{ coordinate0={x，y，z，r}， coordinate1={x，y，z，r}}

4.15 UDP

表 4.52 创建 UDP 网络指令

原型	err, socket = UDPCreate(*isServer*, *IP*, *port*)
描述	创建UDP网络 仅支持单连接

参数	isServer：是否创建服务器。0：表示创建客户端；1：表示创建服务端 IP：服务端IP地址，需与客户端IP地址在同一网段，且不冲突 port：服务端端口 机械臂作为服务端时，“port”不能设置为502或8080，否则会与Modbus默认端口或流水线跟踪中使用的端口冲突，导致创建UDP网络失败
返回	err： 0：创建UDP网络成功 1：创建UDP网络识别 socket：创建的socket对象
示例	请参见程序 4.3和程序 4.4

表 4.53 UDP 接收数据指令

原型	err, Recbuf = UDPRead(*socket*, *timeout*, *type*)
描述	机械臂作为客户端接收来自服务端的数据 机械臂作为服务端时接收来自客户端的数据
参数	socket：socket对象 timeout：超时时间，单位：秒。若timeout为0或者不设置，则该指令为阻塞式读取，即接收完数据后程序才往下执行；若不为0，则超过设定的timeout后，程序继续往下执行，即不考虑数据是否接收完，继续往下执行 type：缓存类型。若设置为空，则RecBuf缓存格式为table形式，如果设置为“string”，则RecBuf缓存字符串形式
返回	err： 0：接收数据成功 1：接收数据失败 Recbuf：接收数据缓存区
示例	请参见程序 4.3和程序 4.4

表 4.54 UDP 发送数据指令

原型	UDPWrite(*socket*, *buf*, *timeout*)
描述	机械臂作为客户端时发送数据给服务端 机械臂作为服务端时发送数据给客户端

参数	socket：socket对象 buf：发送的数据 timeout：超时时间，单位：秒。若timeout为0或者不设置，则该指令为阻塞式读取，即发送完数据后程序才往下执行；若不为0，则超过设定的timeout后，程序继续往下执行，即不考虑数据是否发送完，继续往下执行
返回	0：发送数据成功 1：发送数据失败
示例	请参见程序 4.3和程序 4.4

⚠注意

当前UDP连接仅支持连接单个客户端。连接时请启动服务端后再连接客户端；断开连接时请先关闭客户端，以免服务端端口未及时释放导致再次连接时连接失败。

程序 4.3 UDP 服务端示例

```
local ip="192.168.5.1"                          //机械臂的 IP 地址作为服务端的 IP 地址
local port=6201                                 //服务端端口
local err=0
local socket=0
err, socket = UDPCreate(true, ip, port)
if err == 0 then
    local RecBuf
    while true do
        UDPWrite(socket, "udp server test")     //服务端发送数据至客户端
        err, RecBuf = UDPRead(socket, 0)        //服务端接收客户端发送的数据
        if err == 0 then
            Go(P1)                              //服务端接收到数据后开始执行运动指令
            Go(P2)
            print(buf)
        else
            print("Read error ".. err)
            break;
        end
    end
else
    print("Create failed ".. err)
end
```

程序 4.4 UDP 客户端示例

```
local ip="192.168.1.25"                                    //外部设备的 IP 地址作为服务端的 IP 地址
local port=6200                                            //服务端端口
local err=0
local socket=0

err, socket = UDPCreate(false, ip, port)
if err == 0 then
    local buf
    while true do
        UDPWrite(socket, "udp client test")                //客户端发送数据至服务端
        UDPWrite(socket, {0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07})
        err, RecBuf = UDPRead(socket, 0)                   //客户端接收服务端发送的数据
        if err == 0 then
            Go(P1)                                         //客户端接收到数据后开始执行运动指令
            Go(P2)
            print(buf)
        else
            print("Read error ".. err)
            break
        end
    end
else
    print("Create failed ".. err)
end
```

4.16 Modbus

4.16.1 Modbus 寄存器说明

Modbus是一种串行通信协议，通过此协议，Dobot机器人控制系统可与外部设备进行通信。其中，外部设备如PLC作为Modbus主站，Dobot机器人控制系统作为Modbus从站。

Modbus协议的数据主要分为四类：线圈状态、离散量输入、输入寄存器、保持寄存器，基于Dobot机器人控制系统的内存空间，我们定义了线圈、触点（离散输入）、输入、保持寄存器，用于外部设备与机器人控制系统进行数据交互。其中，每个寄存器地址为4096个。详细说明如下所示。

- 线圈寄存器

表 4.55 线圈寄存器地址说明

线圈寄存器地址（PLC为例）	线圈寄存器地址（Dobot机器人控制系统）	数据类型	说明
00001	0	位	启动
00002	1	位	暂停
00003	2	位	继续
00004	3	位	停止
00005	4	位	急停
00006	5	位	清除报警
00007~0999	6~998	位	系统预留
01001~04096	999~4095	位	用户自定义

- 触点寄存器

表 4.56 触点寄存器地址说明

触点寄存器地址（PLC为例）	触点寄存器地址（Dobot机器人控制系统）	数据类型	说明
10001	0	位	自动退出
10002	1	位	准备好状态
10003	2	位	暂停状态
10004	3	位	运行状态
10005	4	位	报警状态
10006~10999	5~998	位	系统预留
11000~14096	999~4095	位	用户自定义

- 输入寄存器

表 4.57 输入寄存器地址说明

输入寄存器地址（PLC为例）	输入寄存器地址（Dobot机器人控制系统）	数据类型	说明
30001~34096	0~4095	字	系统预留

 版权所有 © 越疆科技有限公司

- 保持寄存器

表 4.58 保持寄存器地址说明

保持寄存器地址（PLC为例）	保持寄存器地址（Dobot机器人控制系统）	数据类型	说明
40001~41000	0~999	字	系统预留
41001~44095	1000~4095	字	用户自定义

4.16.2 指令说明

表 4.59 读线圈功能指令

原型	GetCoils(*addr*, *count*)
描述	读取Modbus从站线圈寄存器地址的值
参数	addr：线圈寄存器起始地址。取值范围：0~4095 count：连续读取线圈寄存器的地址个数。取值范围：0~ 4096 - addr
返回	线圈寄存器地址的值，存储在table中。table中第一个值对应线圈寄存器起始地址的值 返回的数据类型：位
示例	从地址0开始连续读取5个线圈的值 Coils = GetCoils(0,5) 返回结果： Coils={1,0,0,0,0} 结合表 4.55，表示机械臂处于启动状态

表 4.60 写线圈功能指令

原型	SetCoils(*addr*, *count*, *table*)
描述	设置Modbus从站线圈寄存器地址的值 对于Modbus从站线圈寄存器地址0~5，该指令不可用
参数	addr：线圈寄存器起始地址。取值范围：6~4095 count：连续写入线圈寄存器的地址个数。取值范围：0~ 4096 - addr table：线圈寄存器地址的值。数据类型：位
返回	无
示例	从地址1024开始，连续写5个线圈 local Coils = {0,1,1,1,0} setCoils(1024, #coils, coils)

表 4.61 读离散输入功能指令

原型	GetInBits(*addr*, *count*)
描述	读取Modbus从站触点寄存器地址的值
参数	addr：触点寄存器起始地址地址。取值范围：0 ~ 4095 count：连续读取触点寄存器的地址个数。取值范围： 0 ~ 4096-addr
返回	触点寄存器地址的值，存储在table中。table中第一个值对应线圈寄存器起始地址的值 返回的数据类型：位
示例	从地址0开始连续读取5个地址的值 inBits = GetInBits(0,5) 返回结果： inBits = {0,0,0,1,0} 结合表 4.55，表示机械臂处于运行状态

表 4.62 读输入寄存器功能指令

原型	GetinRegs(*addr*, *count*, *type*)
描述	按照指定的数据类型，读取Modbus从站输入寄存器地址的值
参数	addr：输入寄存器起始地址。取值范围：0 ~ 4095 count：读取输入寄存器的地址个数。取值范围：0 ~4096-addr type：数据类型 • 如果为空，默认读取16位无符号整数（2个字节，占用1个寄存器） • “U16”：读取16位无符号整数（2个字节，占用1个寄存器） • “U32”：读取32位无符号整数（4个字节，占用2个寄存器） • “F32”：读取32位单浮点数（4个字节，占用2个寄存器） • “F64”：读取64位双精度浮点数（8个字节，占用4个寄存器）
返回	输入寄存器地址的值，存储在table中。table中第一个值对应线圈寄存器起始地址的值
示例	示例1：从地址2048开始读取一个16位无符号整数 data = GetInRegs(2048,1) 示例2：从地址2048开始读取一个32位无符号整数 data = GetInRegs(2048, 1, “U32”)

表 4.63 读保持寄存器功能指令

原型	GetHoldRegs(*addr*, *count*, *type*)
描述	按照指定的数据类型，读取Modbus从站保持寄存器地址的值

参数	addr：保持寄存器的起始地址。取值范围：0 ~ 4095 count：读取保持寄存器的地址个数。取值范围：0 ~4096-addr type：数据类型 • 如果为空，默认读取16位无符号整数（2个字节，占用1个寄存器） • “U16”：读取16位无符号整数（2个字节，占用1个寄存器） • “U32”：读取32位无符号整数（4个字节，占用2个寄存器） • “F32”：读取32位单精度浮点数（4个字节，占用2个寄存器） • “F64”：读取64位双精度浮点数（8个字节，占用4个寄存器）
返回	保持寄存器地址的值，存储在table中。table中第一个值对应线圈寄存器起始地址的值
示例	示例1：从地址2048开始读取一个16位无符号整数 data = GetHoldRegs(2048,1) 示例2：从地址2048开始读取一个32位无符号整数 data = GetInRegs(2048, 1, “U32”)

表 4.64 写保持寄存器功能指令

原型	SetHoldRegs(*addr*, *count*, *table*, *type*)
描述	设置保持寄存器地址的值
参数	addr：保持寄存器的起始地址。取值范围：0 ~ 4095 count：写入保持寄存器的地址个数。取值范围：0 ~4096-addr table：保持寄存器地址的值 type：数据类型 • 如果为空，默认写入16位无符号整数（2个字节，占用1个寄存器） • “U16”：写入16位无符号整数（2个字节，占用1个寄存器） • “U32”：写入32位无符号整数（4个字节，占用2个寄存器） • “F32”：写入32位单精度浮点数（4个字节，占用2个寄存器） • “F64”：写入64位双精度浮点数（8个字节，占用4个寄存器）
返回	无
示例	示例 1：从地址 2048 开始，写入一个 16 位无符号整数 local data = {6000} SetHoldRegs(2048, #data, data, “U16”) 示例2：从地址2048开始，写入一个64位双精度浮点数 local data = {95.32105} SetHoldRegs(2048, #data, data, “F64”)