九众九机器人
JZJRT

国家工信部

重点产品+工艺一条龙

应用示范参与单位

国资委

系统操作手册

System Operation Manual

目录

第1章 机器人的坐标系与轴操作

本章主要说明本操作系统的坐标系与轴操作的有关事项。

1.1 控制组与坐标系

1.1.1 坐标系

对本体进行轴操作时，其坐标系有以下几种形式，如下图所示。

- 关节坐标系：

单独运动机器人的各个关节轴。

- 直角坐标系：

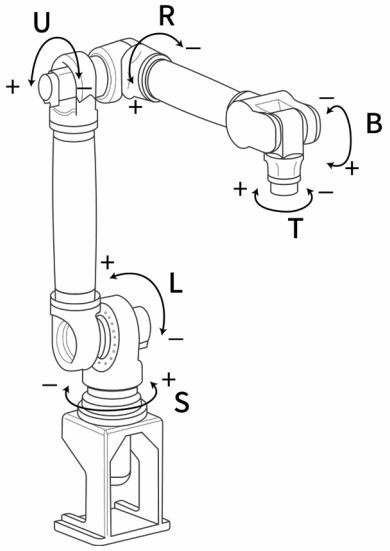
机器人前端沿基座的 X 轴、Y 轴、Z 轴平行运动。A、B、C 分别为绕 X、Y、Z 轴转动。本系统使用的欧拉角顺序为 X'Y'Z'，固定角顺序为 ZYX。

- 工具坐标系：

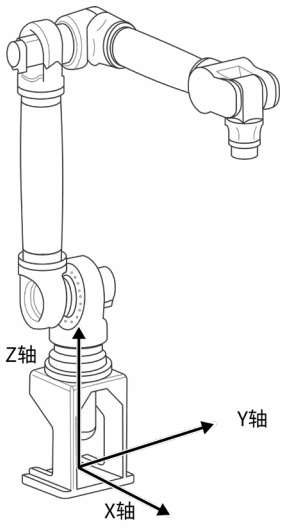
工具坐标系把机器人腕部工具的有效方向作为 Z 轴，把坐标系原点定义在工具的尖端点，本体尖端点根据坐标平行运动。TA、TB、TC 分别为绕 TX、TY、TZ 轴转动。

- 用户坐标系：

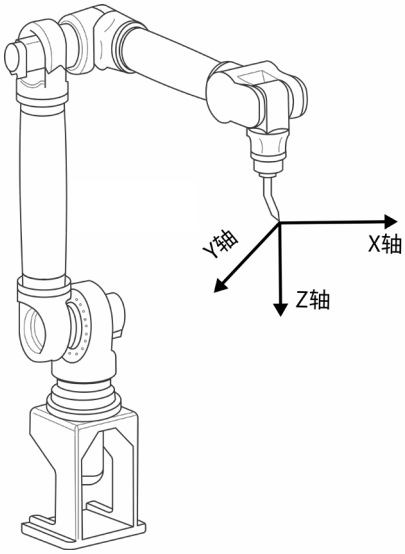
XYZ 直角坐标在任意位置定义。本体尖端点根据坐标平行运动。



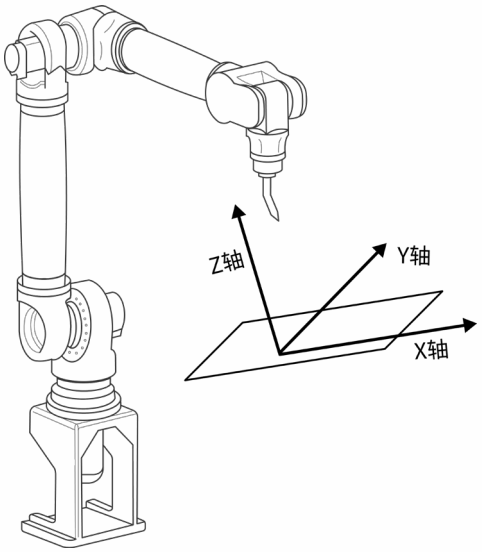
关节坐标系



直角坐标系



工具坐标系



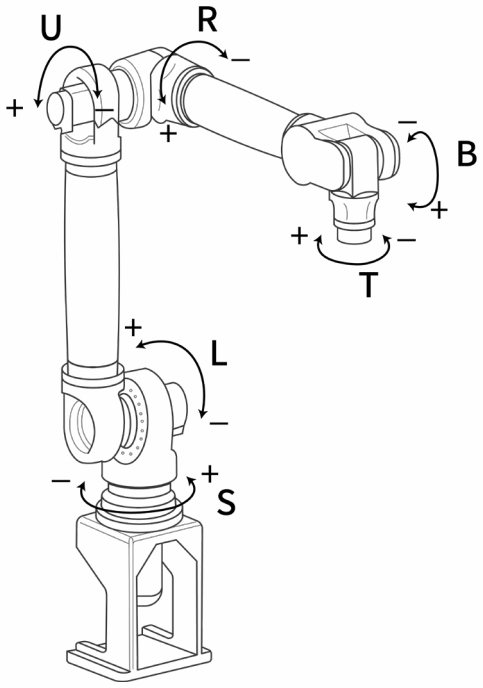
用户坐标系

1.2 坐标系与轴操作

1.2.1 关节坐标系

在关节坐标系，机器人各个轴可单独动作。

当按下机器人没有的轴操作键时，不做任何动作。

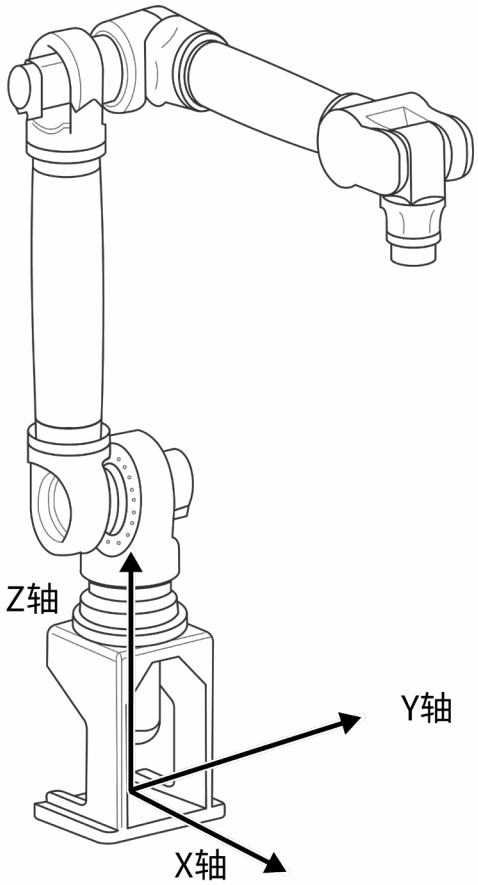


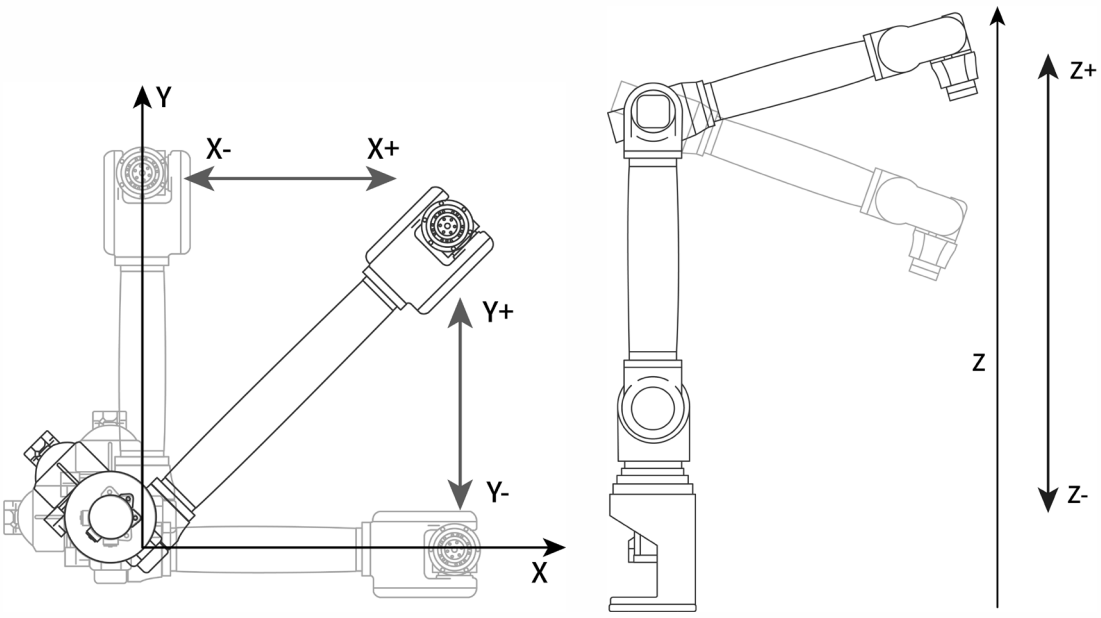
1.2.1.1 关节坐标系的轴操作

轴名称		轴操作	动作
基本轴	S 轴	S+/S-	本体左右旋转
	L 轴	L+/L-	下臂前后运动
	U 轴	U+/U-	上臂上下运动
腕部轴	R 轴	R+/R-	手腕旋转
	B 轴	B+/B-	手腕上下运动
	T 轴	T+/T-	手腕旋转

1.2.2 直角坐标系

机器人在直角坐标系，与本体轴 X、Y、Z 轴平行运动，如下图所示。





1.2.2.1 直角坐标系的轴操作

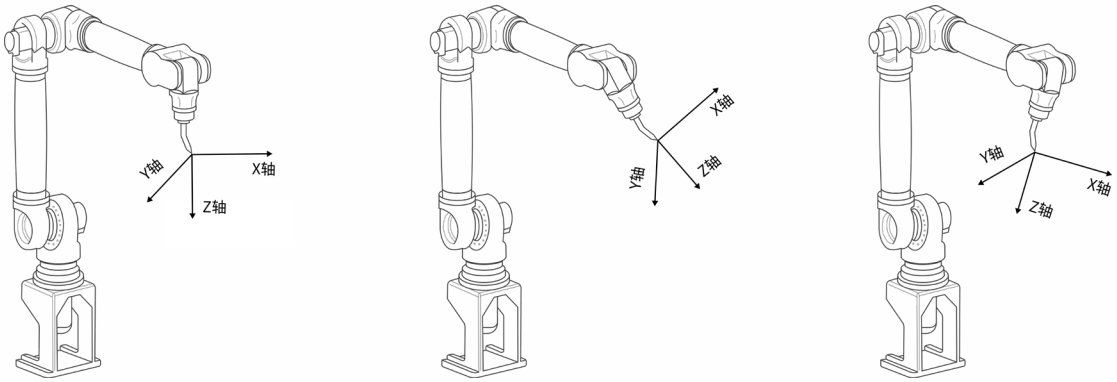
轴名称		轴操作	动作
基本轴	X 轴	X+/X-	沿 X 轴平行移动
	Y 轴	Y+/Y-	沿 Y 轴平行移动
	Z 轴	Z+/Z-	沿 Z 轴平行移动
姿态轴	A 轴	A+/A-	绕 X 轴旋转
	B 轴	B+/B-	绕 Y 轴旋转
	C 轴	C+/C-	绕 Z 轴旋转

1.2.3 工具坐标系

在工具坐标系，机器人沿定义在工具尖端点的 X、Z、Y 轴平行运动。

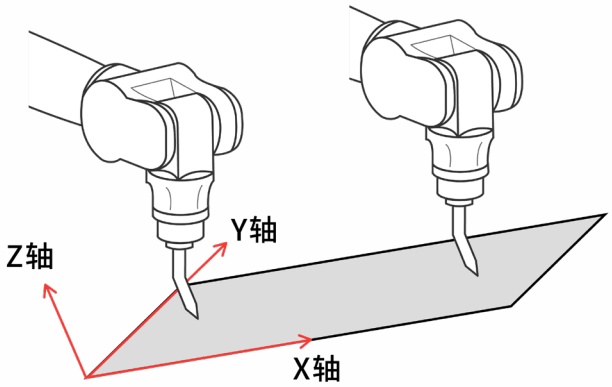
工具坐标把安装在机器人腕部法兰盘上的工具有效方向作为 Z 轴，把坐标定义在工具尖端点。

为此，工具坐标轴的方向随腕部的动作而变化。如下图所示。



工具坐标的运动不受机器人位置或姿势的变化影响，主要以工具的有效方向为基准进行运动。

所以，工具坐标运动最适合在工具姿势始终与工件保持不变、平行移动的应用中使用。如下图所示。

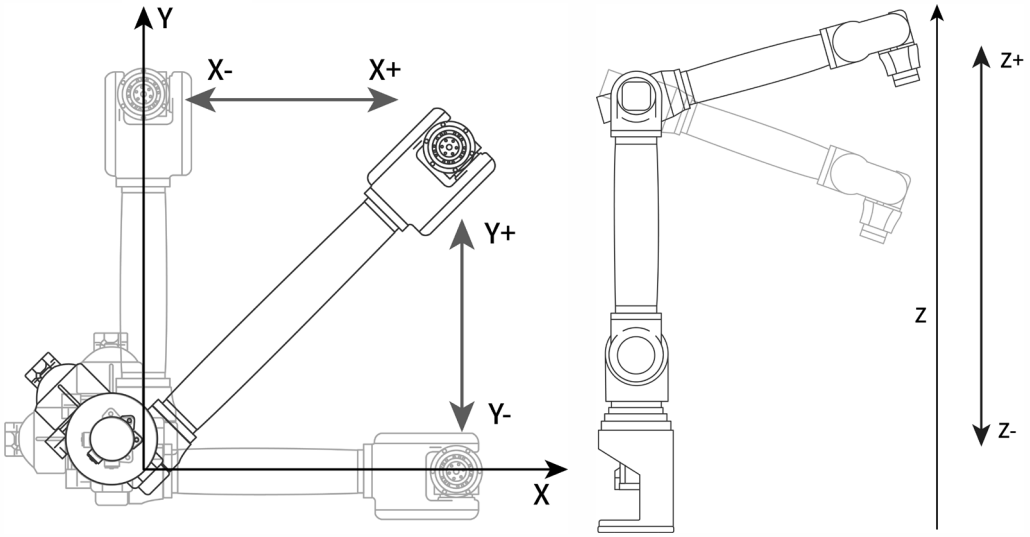


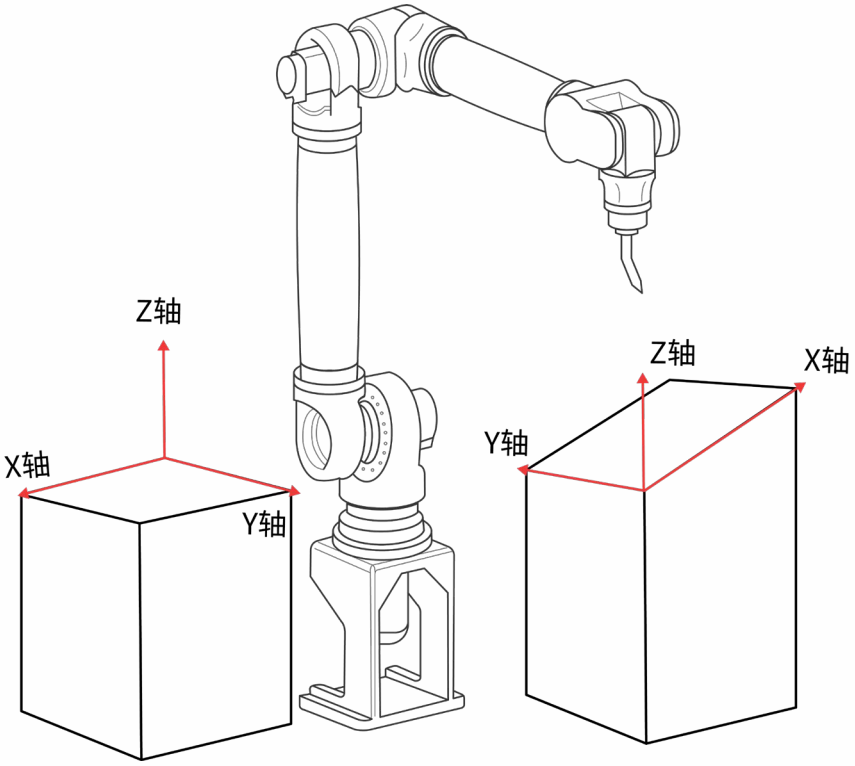
1.2.3.1 工具坐标系的轴操作

轴名称		轴操作	动作
基本轴	TX 轴	TX+/TX-	沿 TX 轴平行移动
	TY 轴	TY+/TY-	沿 TY 轴平行移动
	TZ 轴	TZ+/TZ-	沿 TZ 轴平行移动
姿态轴	TA 轴	TA+/TA-	绕 TX 轴旋转
	TB 轴	TB+/TB-	绕 TY 轴旋转
	TC 轴	TC+/TC-	绕 TZ 轴旋转

1.2.4 用户坐标系

在用户坐标系，在机器人动作范围的任意位置，设定任意角度的 X、Y、Z 轴，机器人与设定的这些轴平行移动。如下图所示。





1.2.4.1 用户坐标系的轴操作

轴名称		轴操作	动作
基本轴	UX 轴	UX+/UX-	沿 UX 轴平行移动
	UY 轴	UY+/UY-	沿 UY 轴平行移动
	UZ 轴	UZ+/UZ-	沿 UZ 轴平行移动
姿态轴	UA 轴	UA+/UA-	绕 UX 轴旋转
	UB 轴	UB+/UB-	绕 UY 轴旋转
	UC 轴	UC+/UC-	绕 UZ 轴旋转

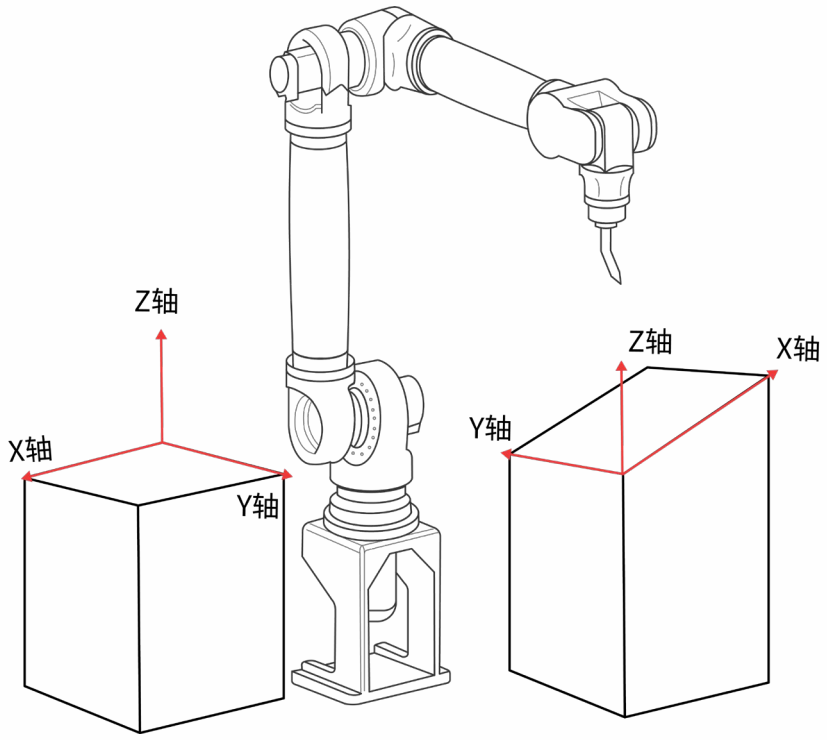
1.2.4.2 用户坐标系的使用举例

通过用户坐标的使用，可使各种示教操作更为简单。

以下通过几个例子加以说明。

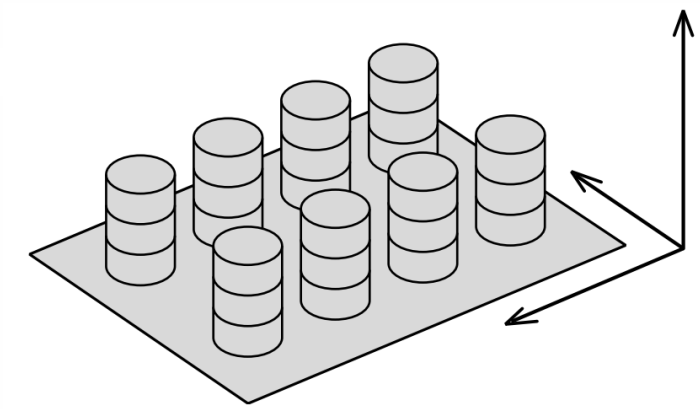
有多个夹具台时:

若使用各夹具台设定的用户坐标，可使手动操作更为简单。



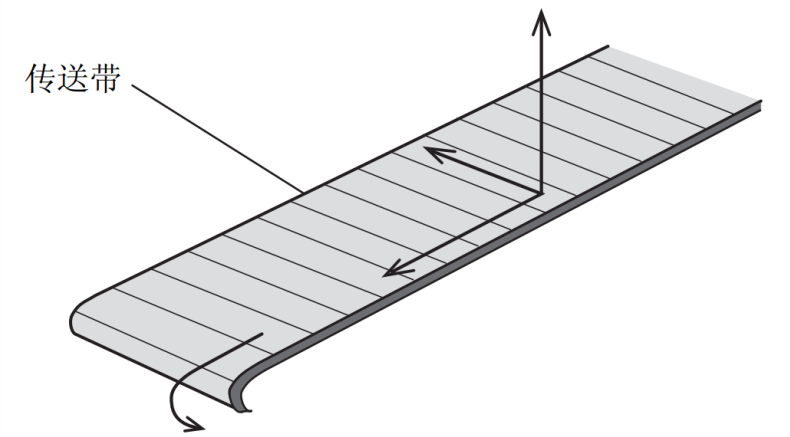
当从事排列、码放作业时：

若将用户坐标设定在托盘上，那么设定平行移动时的位移增加值，就变得更为简单。



与传送带同步运行时：

指定传送带的运动方向。



1.3 外部轴

使用外部轴按钮切换到外部轴后可以进行点动示教外部轴；外部轴仅支持关节点动。

轴名称	轴操作	动作
O1 轴	J1+/J1-	外部轴 1 轴旋转运动
O2 轴	J2+/J2-	外部轴 2 轴旋转运动
O3 轴	J3+/J3-	外部轴 3 轴旋转运动
O4 轴	J4+/J4-	外部轴 4 轴旋转运动
O5 轴	J5+/J5-	外部轴 5 轴旋转运动

第2章 示教器按键与界面简介

2.1 T20 示教器物理按键

左侧

	切换到用户界面
	切换到设置界面
	切换到变量界面
	切换到工程界面
	切换到程序界面
	切换当前示教使用的坐标系
	切换到日志界面

下侧

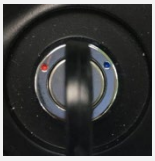
	伺服报错后清错。（仅在示教模式下有效）
	回 复位点按钮
	切换当前伺服状态
	切换当前机器人。（仅多机模式时可用）
	在当前机器人与外部轴之间切换。（仅在有外部轴时可用）
	切换示教模式下单步运行程序时为顺序执行还是逆序执行。

	在示教模式下单步运行程序。
	降低示教或运行速度。
	增大示教或运行速度。

右侧

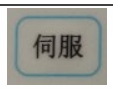
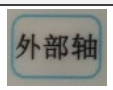
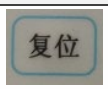
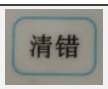
	运行模式下暂停程序
	运行模式下开始程序
	示教时对应轴负方向运行
	示教时对应轴正方向运行
	回到零点

钥匙开关

	左边，切换到示教模式
	中间，切换到运行模式
	右边，切换到远程模式

2.2 T30 示教器物理按键

左侧

	切换当前伺服状态
	切换当前机器人。（仅多机模式时可用）
	在当前机器人与外部轴之间切换。（仅在有外部轴时可用）
	回零点按键
	回复位点按键
	伺服报错后清错。（仅在示教模式下有效）
	预留

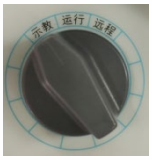
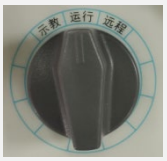
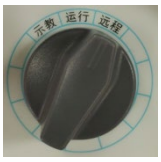
下侧

	切换示教模式下单步运行程序时为顺序执行还是逆序执行。
	在示教模式下单步运行程序。
	降低示教或运行速度。
	增大示教或运行速度。
	切换工具手（预留）
	切换示教模式下单步运行程序时为顺序执行还是逆序执行。

右侧

	运行模式下暂停程序
	运行模式下开始程序
	示教时对应轴负方向运行
	示教时对应轴正方向运行

钥匙开关

	左边，切换到示教模式
	中间，切换到运行模式
	右边，切换到远程模式

急停按钮

	按下急停

滚轮旋钮

	程序界面选转切换上一行、下一行

2.3 操作系统简介

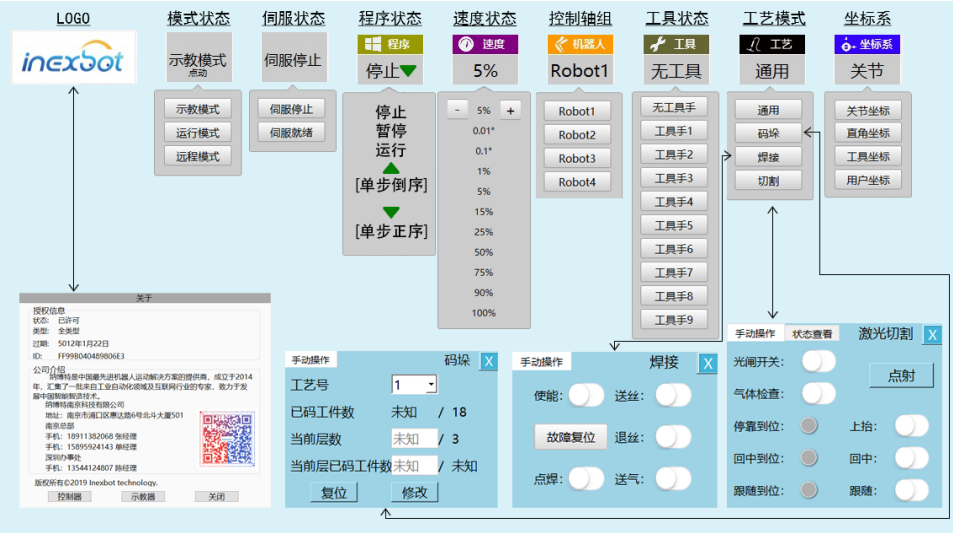
2.3.1 基本说明

本节主要对程序界面各个部分进行 概述。程序左侧为功能键其他功能如表

管理员	打开管理员/技术员/操作员切换界面
设置	打开机器人功能设置界面
工艺	打开机器人工艺选择界面
X=/变量	打开机器人变量设置界面
状态	打开机器人状态查看界面
工程	打开工程预览界面
程序	打开程序指令界面
日志	打开报错日志界面
监控	打开机器人监控显示界面
12:30 星期四 2016/08/30	日期和时间显示

2.3.2 状态介绍

程序顶部为状态栏，显示机器人的各个状态。



模式状态　通过旋转【模式选择钥匙】来切换，分别有示教模式、远程模式、运行模式。

伺服状态　启动程序后按下【Mot】按键来从“伺服停止”状态切换到“伺服就绪”状态。
当在示教模式按下【 DEADMAN】按键或在“再现模式”或“循环模式”运行程序时，伺服状态切换为“伺服运行”状态。

程序状态　程序的运行状态，当在“示教模式”以 STEP 单步运行或在“再现模式”、“循环模式”下运行程序时，程序状态切换为“运行”状态

点动速度　通过按下示教器底部的【V-】、【V+】来增大或减小点动速度

- 速度增大：按动示教器底部的【V+】(速度增加)按钮，每按一次，点动速度按以下顺序变化：微动 1%→微动 2%→低 5%→低 10%—中 25%→中 50%→高 75%高 100%
- 速度减小：按动示教盒底部的【V】(速度减小)按钮，每按一次，点动速度按以下顺序变化：高 100%→高 75%→中 50%→中 25%低 10%→低 5%→微动 2%微动 1%

机器人状态　通过按下示教器底部的【Rob】按键来切换。
分别为“ Robot a”和“ Robot b”两个状态。

工具状态　通过按下示教器底部的【Jog】按键来切换。
分别为“工具 1”、“工具 2”、“工具 3”、“工具 4”、“工具 5”、“工具 6”、“工具 7”、“工具 8”、“工具 9”三个状态。

工艺模式　【示教状态下】通过通过手动来切换。
分别为“通用”、“焊接”、“码垛”、“激光切割”四个状态。

坐标系　通过按下示教器左侧的【坐标系切换】按键来切换。
分别有“关节坐标系”、“直角坐标系”、“工具坐标系”、“用户坐标系”四种坐标系。

第3章 机器人示教与运行

本章将会说明 NRC 机器人控制系统的具体编程步骤，以及各控制指令的详细说明。

3.1 机器人准备

3.1.1 开机与安全确认

本节主要讲述进行示教操作前的开机以及确认安全措施完备的方法。

3.1.1.1 开机

操作步骤：

1. 检查伺服、控制器、示教盒各部件连接线是否已连接完好。
2. 把机柜面板上的主电源开关旋转到接通（ON）的位置，主电源接通。
3. 按下机柜面板上的绿色伺服启动按钮。

3.1.1.2 安全确认

出于安全上的考虑，示教前请确认急停按钮是否正常。

急停按钮的使用确认：

在机器人使用前，请分别对控制柜、示教盒上的急停按钮进行确认，按下时，伺服电源是否断开

1. 按控制柜及示教盒上的急停按钮；
2. 确认伺服电源关闭，示教器显示伺服报错，控制柜伺服报错灯亮；
3. 清除伺服错误，控制柜伺服报错灯灭，示教器上显示“伺服停止”；
4. 确认正常后，按示教盒上的 “MOT”键，使伺服处于伺服准备状态；

3.1.2 示教器准备

待示教器开机且确认伺服无报错后，确认示教器在示教模式下，如没有则旋转模式选择钥匙，将示教器切换到示教模式下。按下示教器上的【MOT】（伺服准备）按，此时程序界

面上方的“伺服状态”一栏显示为“伺服就绪”且闪烁伺服就绪。*只有在“伺服就绪”状态机器人才可以使能!*

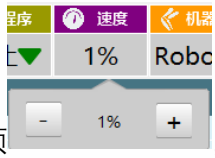
轻按示教器背后的【DEADMAN】按键，此时听到机器人上电的声音，且“伺服状态”一栏显示为绿色的“伺服运行”，表示伺服电源成功接通。

3.2 点动操作

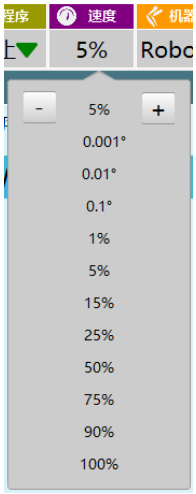
本节主要讲述示教模式下利用示教器物理按键实现手动操作的相关事项。包括坐标系的定义及其设置，手动操作的方法，速度设置及手动操作时各状态的确认。若要掌握熟练需要多次实际操作运用。

3.2.1 示教速度调节

在示教模式下，修改手动操作机器人运动速度，按手持操作示教器上【V+】（速度增加）键或【V-】（速度减小）键，每按一次，手动速度按以下顺序变化，通过状态区的速度显示来确认。

也可以点击状态栏中的速度一项 

，会弹出下拉菜单，点击“-”和“+”同样能够加减速度。点击中间的数字会弹出速度选项，可以快速选择几个常用速度。



速度增大：按动示教器底部的【V+】（速度增加）按钮，每按一次，手动操作速度按以下顺序变化：

寸动 0.001°→寸动 0.01°→寸动 0.1°→1\%→5\%→10\%→速度增加 5%，直到 100%

速度减小：按动示教盒底部的【V-】（速度减小）按钮，每按一次，手动操作速度按以下顺序变化：

高 100\%→每次减 5%→低 5\%→微动 1\%→寸动 0.1°→寸动 0.01°→寸动 0.001°

寸动：寸动速度在关节坐标系下有 0.01°和 0.1°两档。在直角、工具、用户坐标系下有 0.1mm、1mm 两档。

示教速度是按百分比来的，其实际速度为点动最大速度*状态栏中的百分比。点动最大速度在设置-机器人参数-点动速度界面中设置，*具体参数说明与设置方法请见机器人设置一章。*

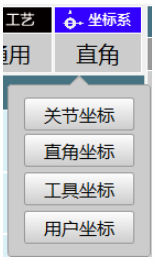
3.2.2 坐标系说明与切换

在本产品中含有四种坐标系，分别为关节坐标系、直角坐标系、工具坐标系和用户坐标系。

- 关节坐标系所有点位均为机器人关节轴相对于轴机械零点的角度值。
- 直角坐标系又叫“基坐标系”，其所有点位均为机器人末梢（法兰中心）相对于机器人基座中心的坐标值（单位 mm）；
- 工具坐标系所有点位均为机器人所带工具末梢（TCP 点）相对于机器人基座中心的坐标值（单位 mm）。*其定义和使用方法请见工具手与用户坐标一章;*
- 用户坐标系又叫“工件坐标系”，其所有点位均为机器人所带工具末梢（未带工具时为其法兰中心）相对用户坐标系原点的坐标值（单位 mm）。*其定义和使用方法请见工具手与用户坐标一章。*

在示教模式下，按动示教盒下方物理按键区的【坐标系切换】按键，每按一次此键，坐标系按以下顺序切换，通过顶部状态栏的显示来确认。

也可以点击状态栏的坐标系一栏，即可弹出坐标系选择菜单，点击对应坐标系即可切换。



关节→直角→工具→用户

3.2.3 点动操作

若要进行机器人的点动操作，具体为以下步骤:

1. 开机。
2. 检查急停按钮是否完好，是否按下。
3. 按动示教盒的 MOT 按键，确定伺服状态为“伺服准备”。
4. 选择需要使用的坐标系。
5. 调整到合适的速度。
6. 按动示教器的【DEADMAN】按键（***示教器背后的按钮***），***不松手***。
7. 使用示教器右侧物理按键区的按键操作机器人运动。
8. 松开【DEADMAN】按键。

3.3 程序编写

本节将主要介绍本产品的各项对指令的操作。包括程序的新建、修改、删除、复制和重命名以及指令的插入、修改、删除和复制等操作，以及各指令的具体功能说明，并提供具体示例。***若要熟练掌握需经过多次实际运用***。

3.3.1 程序新建/打开/删除/重命名/复制

用户若要进行程序的插入/修改/删除/复制/重命名等指令相关的操作，需进入程序界面，通过使用底部按钮进行相关操作。

3.3.1.1 新建程序

新建程序需通过点击工程界面底部的【新建】按钮。

新建的程序在选中的程序下面。

相关步骤如下:

1. 进入工程界面;

2. 在弹出的“程序创建”窗口中输入相应的程序名称等参数。

3. 点击底部的【确定】按钮，程序创建成功，并跳转入新建的程序。若想要取消新建程序，则点击【取消】按钮。

注意

程序名称必须为以字母/汉字开头的两位及以上的字符串。

程序名称不能为已有程序的名称。

3.3.1.2 程序打开

用户若要打开已有的作业文件，则需要进行以下步骤：

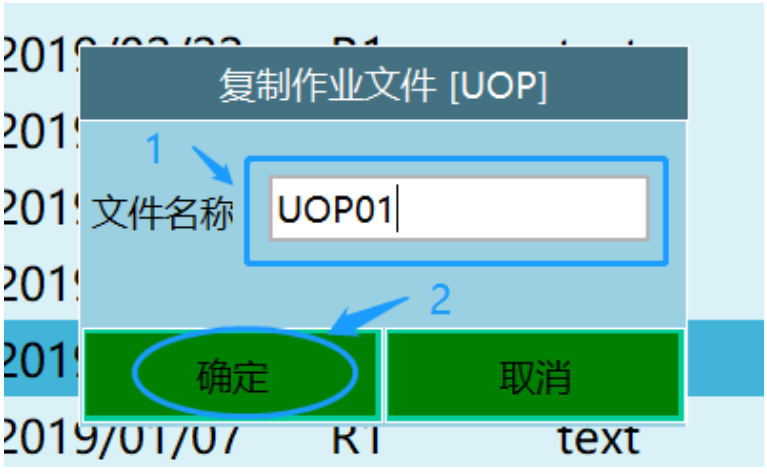
1. 打开“工程”界面；
2. 选中想要打开的程序；
3. 点击底部的【打开】按钮。程序打开成功。

3.3.1.3 程序复制

用户若要复制已有的作业文件（只能整体复制），则需要进行以下步骤：

1. 选中要复制的程序；

2. 点击底部的【操作】按钮，再点击【复制】；



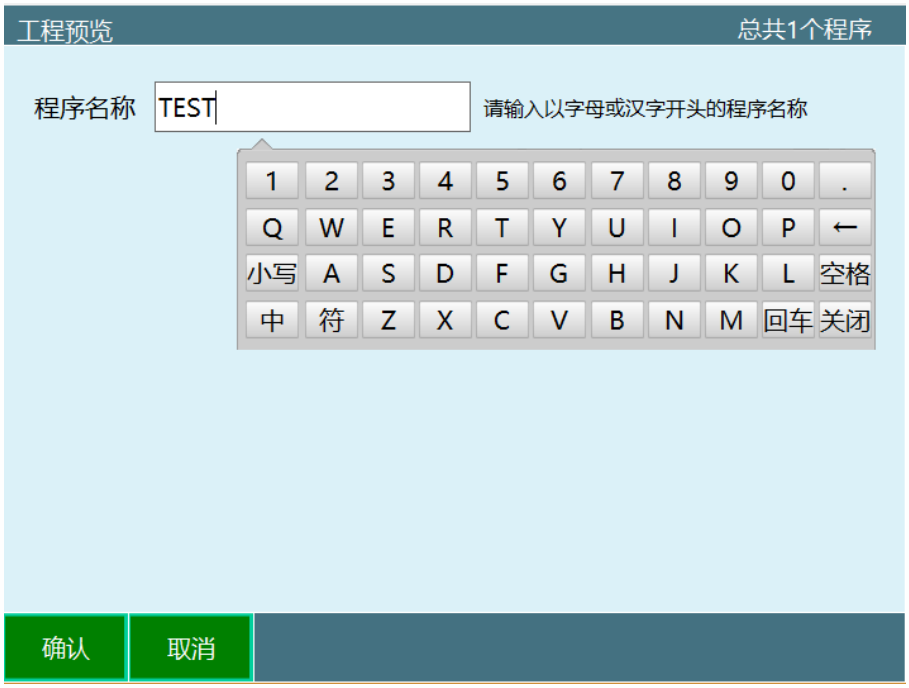
3. 点击【确定】，否则【取消】；您也可以修改文件名称。

3.3.1.4 程序重命名

重命名操作可以修改选中程序的名称。

操作步骤如下：

1. 选中想要重命名的程序。
2. 点击【操作】，再点击【重命名】
3. 在弹出的窗口中输入想要修改的名称。



4. 点击【确定】按钮。若想要取消重命名操作，则点击【取消】按钮。

重命名的程序的程序名不能为已有程序的名称。

3.3.1.5 程序删除

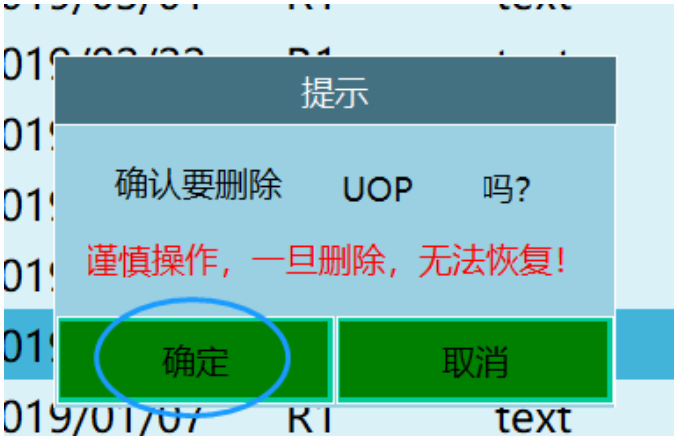
删除操作可以删除选中的程序。

相关操作步骤如下：

1. 选中想要删除的程序。

2. 点击删除按钮；

3. 在弹出的窗口中点击【确定】按钮。若想要取消删除操作，则点击【取消】按钮。



3.3.1.6 批量删除

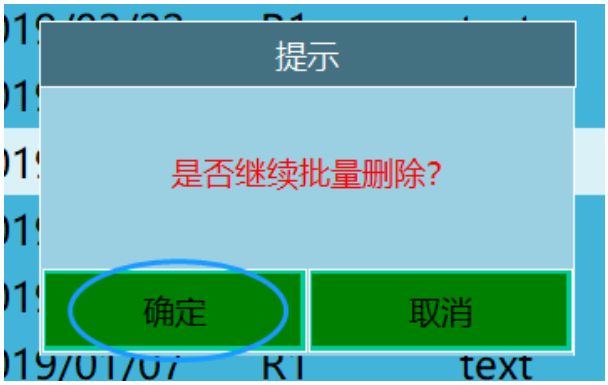
批量删除功能可以一次删除多个程序文件。使用方法如下：

1. 进入工程界面；

2. 点击底部菜单栏的操作-批量删除按钮；

3. 选中需要删除的程序文件（仅能选中当前页的文件，不能进入上一页或下一页），点击全选按钮则选中本页全部程序文件；

4. 点击【确定按钮】按钮后再弹出的确认框中点击【确定】按钮则批量删除成功。



3.3.2 指令操作

用户若要进行指令的插入/修改/删除等指令相关的操作，需进入程序预览界面，通过使用底部按钮进行相关操作。

3.3.2.1 插入

指令的插入需通过使用程序预览界面底部的【指令菜单】按进行相关操作。

插入的指令在选中指令行的下面

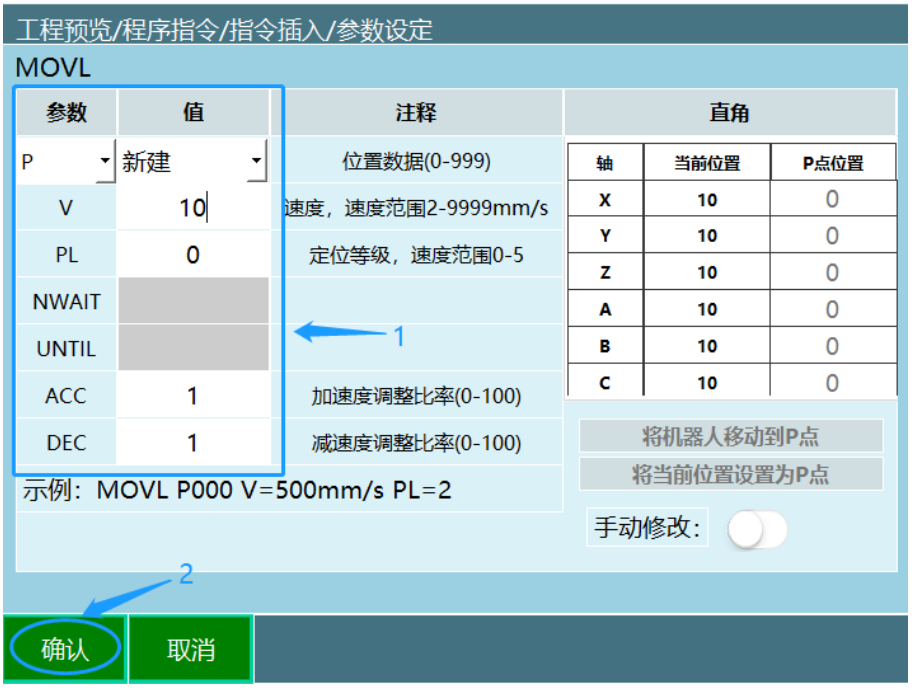
相关步骤如下：

1. 进入程序预览界面；

2. 点击【插入】按钮，弹出指令类型菜单；

3. 点击所需插入指令的指 令类型，例如运动控制类，如图；

4. 点击所需插入的指令，例如 MOVL，如图；



5. 设置所插入指令的相关参数；

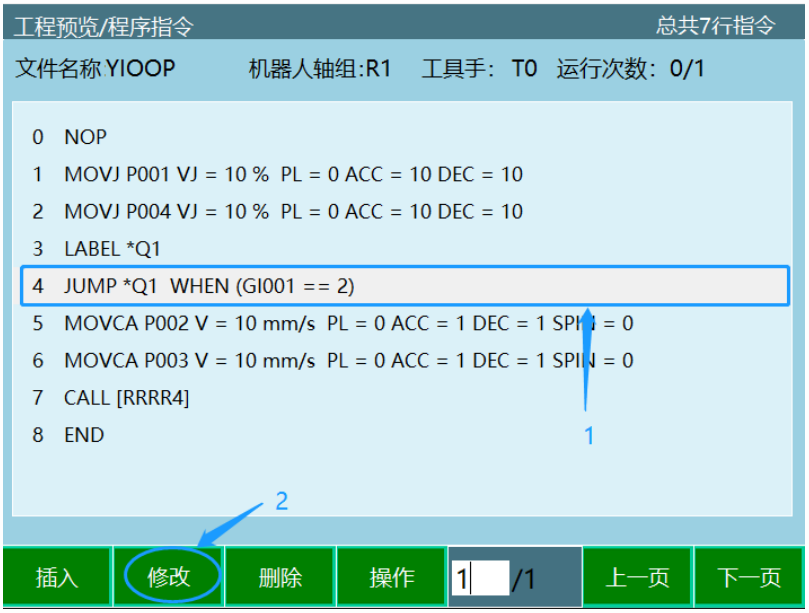
6. 点击程序底部【确认】按钮。

3.3.2.2 指令修改

用户可以通过使用“修改”命令方便地修改已插入指令的相关参数。

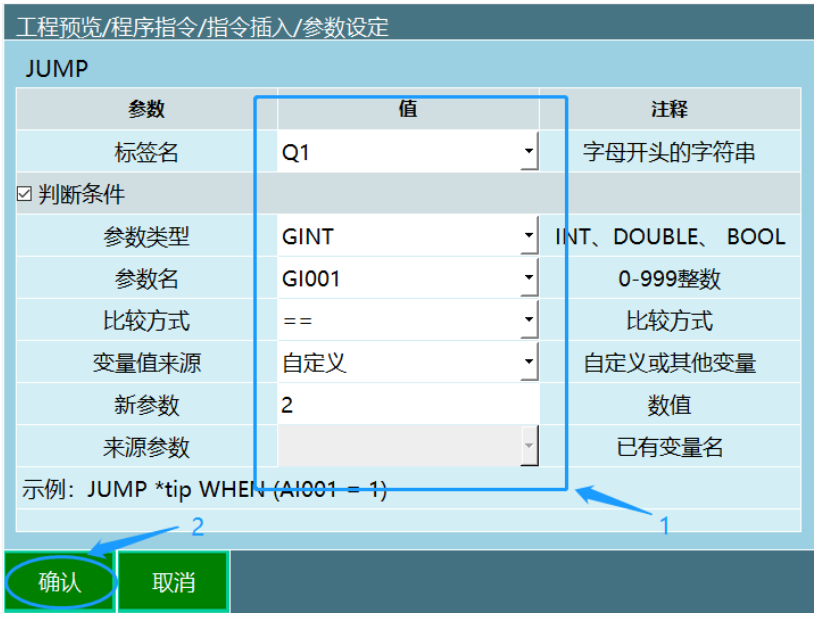
修改指令参数的步骤如下:

1. 选中已插入行（NOP 行和 END 除外）；



2. 点击程序底部的【修改】按钮

3. 修改相关参数;



4. 修改完成后点击底部的【确定】按钮

5. 指令修改成功。

注意

重命名的程序的程序名不能为已有程序的名称。

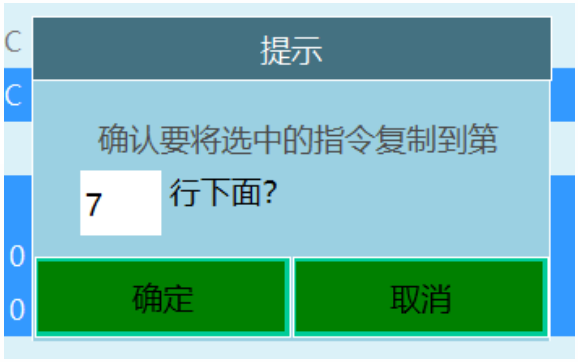
3.3.2.3 批量复制

用户可以通过“批量复制”操作复制需要的指令到指定的地方。步骤如下：

1. 首先点击底部“操作”按钮中的 批量复制；

2. 选择需要的指令；

3. 点击“确认复制”按钮，弹出下图按钮，填写您粘贴的位置即可；



3.3.3 指令说明（指令规范）

本节主要说明各指令的功能及相关参数作用及规范。提供一些具体应用场景示例。

3.3.3.1 运动控制类

运动控制类指令包括 MOVJ、MOVL、MOVC、IMOV、NOVCA、MOVJEXT、MOVLEXT、MOVCEXT 等指令。

在插入这些运动控制类指令时若选中 P 点为新建，则同时会自动新建一个 P 变量，并将当前机器人位置写入该变量。运行该指令则运行到机器人在插入该指令时的位置。

本系统所有运动类指令的目标点均使用位置变量，局部位置变量为 P，全局位置变量为 G，具体位置变量的使用方法请见 *位置变量一章。*

各指令功能及相关参数作用及范围如下：

- **MOVJ**

  在机器人向目标点移动中，在不受轨迹约束的区间使用。

  若用关节插补示教机器人轴，移动命令是 MOVJ。

  处于安全考虑，通常情况下，请用关节插补示教第一步。

  默认的速度为 VJ=10，即 10\%的最高速度。

MOVJ	功能	以关节插补方式向示教位置移动	
	参数	位置数据、 基座轴位置数据、工装轴位置数据。	界面中显示
		VJ=再现速度	VJ：1-100

		PL=定位等级	PL：0~5
		ACC=加速度调整比率	ACC：1-100
		DEC=减速度调整比率	DEC：1-100
		TIME=下一条指令提前执行时间	
	使用示例	MOVJ P001 VJ=10% PL=2 ACC=10 DEV=10	

- **MOVL**

用直线轨迹在直线插补示教的程序点中移动。

若用直线插补示教机器人轴，移动命令是 MOVL。

直线插补常在焊接作业中使用。

使用直线插补时，机器人手腕姿态不变。

MOVL	功能	以直线插补方式向示教位置移动	
	参数	位置数据、 基座轴位置数据、工装轴位置数据。	界面中显示
		V=再现速度	V：2-9999
		PL=定位等级	PL：0~5
		ACC=加速度调整比率	ACC：1-100
		DEC=减速度调整比率	DEC：1-100
	使用示例	MOVL P001 V=100 PL=2 ACC=10 DEV=10	

- **MOVC**

机器人通过圆弧插补示教的 3 个点画圆移动。

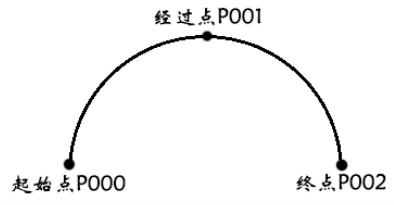
若用圆弧插补示教机器人轴，移动命令是 MOVC。

单一圆弧和连续圆弧的第一个圆弧的起始点只能为 MOVJ 或 MOVL。

- 单一圆弧

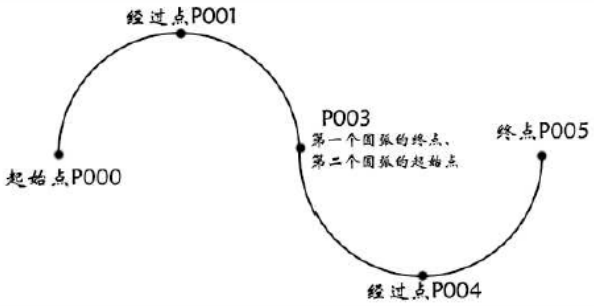
  当圆弧只有一个时，如下表所示，用圆弧插补示教 P1-P3 的 3 个点。

  若用关节插补或直线插补示教进入圆弧前的 P0，则 P0-P1 的轨迹自动成为直线。

	点	插补方式	命令
	P000	关节 直线	MOVJ MOVL
	P001-P002	圆弧	MOVC

■ 连续圆弧

如下表所示，当曲率发生改变的圆弧连续有 2 个以上时，圆弧最终将逐个分离。因此，如图 4 所示，请在前一个圆弧与后一个圆弧的连接点加入关节及直线插补的点。

	点	插补方式	命令
	P000	关节 直线	MOVJ MOVL
	P001-P002	圆弧	MOVC
	P003-P004	圆弧	MOVC

MOVC	功能	圆弧插补方式移动至目标位置。采用三点圆弧法，圆弧前一点为第一点，两个 MOVC 为中间点和目标点。注意：作业文件的第一个运动控制类指令不能为 MOVC。	
	参数	位置数据、 基座轴位置数据、工装轴位置数据。	界面中显示
		V=再现速度	V：2-9999
		PL=定位等级	PL：0~5
		NWALL	
		UNTIL	
		ACC=加速度调整比率	ACC：1-100
		DEC=减速度调整比率	DEC：1-100

	使用示例	MOVC P001 V=100 PL=2 ACC=10 DEV=10

- **IMOV**

IMOV	功能	以关节或直线插补方式从当前位置按照设定的增量值距离移动	
	参数	B=位置数据	BF：基座坐标 RF：机器人坐标 TF：工具坐标 UF：用户坐标
		V=再现速度	V：2-9999
		PL=定位等级	PL：0~5
		用户坐标	显示 B 参数状态
		UNTIL	
		ACC=加速度调整比率	ACC：1-100
		DEC=减速度调整比率	DEC：1-100
	使用示例	IMOV B001 V=100 PL=2 ACC=10 DEV=10	

- **MOVS**

在焊接、切割、熔接、涂底漆等作业时，若使用自由曲线插补，对于不规则曲线工件的示教作业可变得容易。

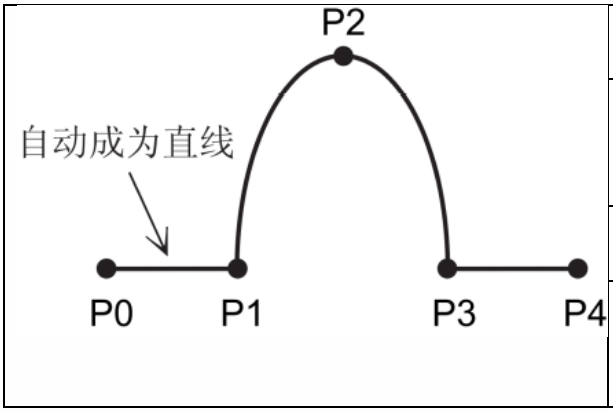
轨迹为通过三个点的抛物线。

若使用自由曲线插补示教机器人轴，则移动命令为 MOVS。

- 单一自由曲线

  如下表所示，用自由曲线插补示教 P1-P3 的 3 个点。

  若使用关节插补或直线插补示教进入自由曲线前的 P0 点，那么 P0-P1 的轨迹自动成为直线。

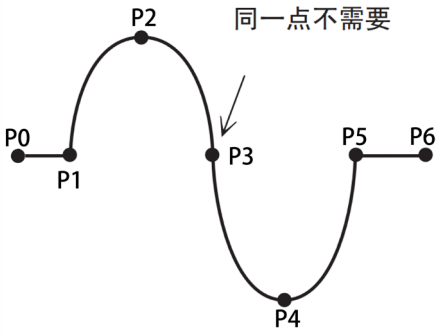


点	插补方式	命令
P000	关节 直线	MOVJ MOVL
P001-P003	自由曲线	MOVS
P004	关节 直线	MOVJ MOVL

■ 连续自由曲线

用重合抛物线合成建立轨迹。

与圆弧插补不同，2 个自由曲线的连接处不能是同一点或不能有其它指令。



点	插补方式	命令
P000	关节 直线	MOVJ MOVL
P001-P005	自由曲线	MOVS
P006	关节 直线	MOVJ MOVL

重合抛物线的情况下建立合成轨迹。

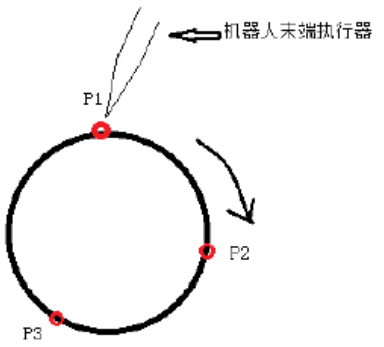
MOVS	功能	以自由曲线插补形式向示教位置移动。	
	参数	位置数据、 基座轴位置数据、工装轴位置数据。	界面中显示
		V=再现速度	V：2-9999
		PL=定位等级	PL：0~5
		NWALL	
		UNTIL	
		ACC=加速度调整比率	ACC：1-100
		DEC=减速度调整比率	DEC：1-100
	使用示例	MOVS P001 V=100 PL=2 ACC=10 DEV=10	

- **MOVCA**

若要示教机器人行走一个完整的圆，移动命令是 MOVCA。

指令插入前提

点击上方状态栏中的“工具”按钮，选中之前标定好的工具手；



插入步骤，共四条指令。

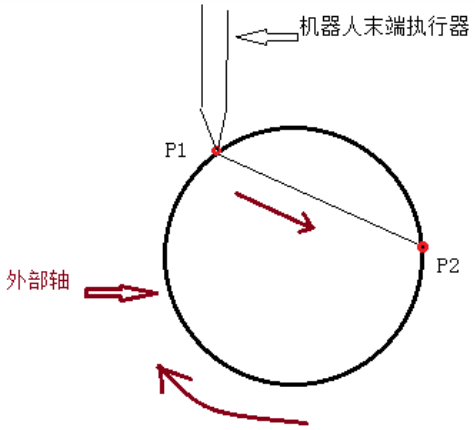
1. 点击插入，点击坐标切换类，选择 SWITCHTOOL，选择之前标定好的工具手号
2. 移动到所要画的圆的任意一个点如图 P1，点击插入，点击运动控制类，选择 movj 或者 movl；
3. 再移动到所要画的圆的任意一个点如图 P2（要不同于第 2 步中的点），点击上方状态栏中的“坐标系”按钮，选中“工具”坐标系，点击插入，点击运动控制类，选择 movca
4. 再移动到所要画的圆的任意一个点如图 P3（要不同于 2,3 步中的点），点击上方状态栏中的“坐标系”按钮，选中“工具”坐标系，点击插入，点击运动控制类，选择 movca

MOVCA	功能	基于三点确定一个圆的原理，画一个圆。采用三点画圆法，圆前一点为第一点，两个 MOVCA 为圆的两个中间点。注意：作业文件的第一个运动控制类指令不能为 MOVCA。	
	参数	位置数据、 基座轴位置数据、工装轴位置数据。	界面中显示
		V=再现速度	V：2-9999
		PL=定位等级	PL：0~5

		NWALL	
		UNTIL	
		ACC=加速度调整比率	ACC：1-100
		DEC=减速度调整比率	DEC：1-100
	使用示例	MOVCA P001 V=100 PL=2 ACC=10 DEV=10	

- **MOVJEXT**

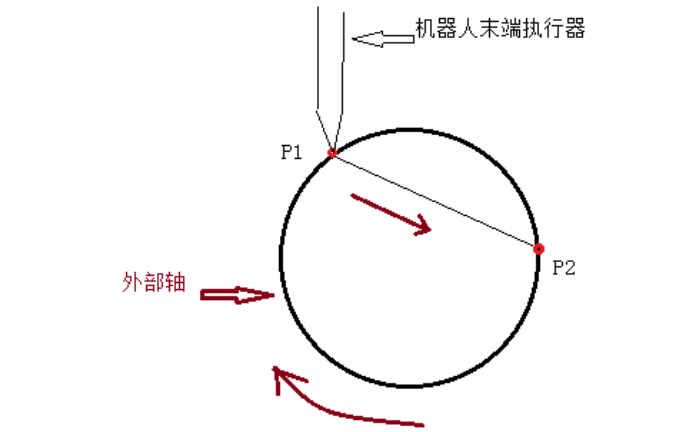
机器人以关节插补方式向示教位置移动，外部轴用用关节差补运动。



MOVJEXT	功能	机器人以关节插补方式向示教位置移动，外部轴用用关节差补运动	
	参数	位置数据、 基座轴位置数据、工装轴位置数据。	界面中显示
		VJ=再现速度	VJ：1-100
		PL=定位等级	PL：0~5
		NWALL	
		UNTIL	
		ACC=加速度调整比率	ACC：1-100
		DEC=减速度调整比率	DEC：1-100
	使用示例	MOVJEXT P001 VJ=10% PL=2 ACC=10 DEV=10	

- **MOVLEXT**

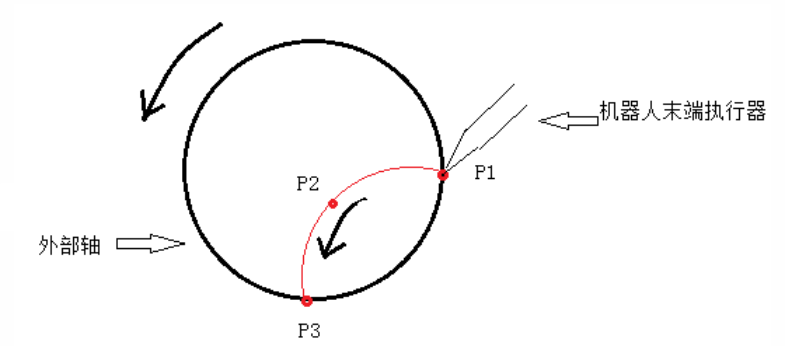
机器人以直线插补的方式向示教位置移动，外部轴用关节差补的方式运动。



MOVL EXT	功能	机器人以直线插补的方式向示教位置移动，外部轴用关节差补运动	
	参数	位置数据、 基座轴位置数据、工装轴位置数据。	界面中显示
		V=再现速度	V：2-9999
		PL=定位等级	PL：0~5
		NWALL	
		UNTIL	
		ACC=加速度调整比率	ACC：1-100
		DEC=减速度调整比率	DEC：1-100
	使用示例	MOVL P001 V=100 PL=2 ACC=10 DEV=10	

- **MOVCEXT**

机器人以圆弧插补方式向示教位置移动，外部轴用用关节差补运动。



MOVC EXT	功能	机器人以圆弧插补的方式向示教位置移动，外部轴用关节差补运动	
	参数	位置数据、 基座轴位置数据、工装轴位置数据。	界面中显示
		V=再现速度	V：2-9999
		PL=定位等级	PL：0~5
		NWALL	
		UNTIL	
		ACC=加速度调整比率	ACC：1-100
		DEC=减速度调整比率	DEC：1-100
	使用示例	MOVCEXT P001 V=100 PL=2 ACC=10 DEV=10	

- **SAMOV**

机器人以关节插补运动到一个设定好的绝对位置

如果你不希望移动某个轴，请在该轴的坐标处留空。（不要填 0）

SAMOV	功能	机器人以关节插补运动到一个设定好的绝对位置	
	参数	B=位置数据	BF：基座坐标 RF：机器人坐标 TF：工具坐标 UF：用户坐标
		V=再现速度	V：2-9999
		PL=定位等级	PL：0~5

		用户坐标	显示 B 参数状态
		UNTIL	
		ACC=加速度调整比率	ACC：1-100
		DEC=减速度调整比率	DEC：1-100
	使用示例	SAMOV B001 V=100 PL=2 ACC=10 DEV=10	

- **SPEED**

SPEED 指令以下的所有运动类指令的运动类指令的运动速度为：指令速度*上方状态栏的速度*SPEED 的百分比。

SPEED	功能	设置全局速度	
	参数	全局速度（%）	速度百分比：1-200
	使用示例	SPEED 200	

3.3.3.2 定时器类

- **TIMER**

定时

TIMER	功能	延迟	
	参数	时间	0-9999s
	使用示例	TIMER=100s	

3.4 程序运行

程序可以在三种模式状态中运行，包括“示教”、“运行”、“远程”，分别对应着“示教模式”、“运行模式”、“远程模式”。用户通过使用示教器右上角的模式选择钥匙可以在三种模式间切换。

3.4.1 示教模式

示教模式下可以完成机器人的点动操作、作业文件编程、系统参数设定等各项操作。其中在作业文件编程的过程中可以使用“STEP”功能来对作业文件进行单步操作。

3.4.1.1 使用单步运行/STEP 进行轨迹确认

用户在选中已插入的指令行后，通过按住【DEADMAN】按键的同时，点击示教器底部的物理按键区中的【STEP】（单步）按键对在编程中的作业文件进行单步操作（机器人运动的过程中不要松开【DEADMAN 按键】）单步操作可以仅运行选中的指令行。

STEP 运行速度=指令速度*上方状态栏的速度比率。

具体步骤如下：

1. 选中要进行单步操作的指令行。
2. 按下【DEADMAN】按键，机器人上电。
3. 按下【STEP】按键，机器人执行选中行的指令，执行完后停止。
4. 选中行自动下移，若要单步运行下一行指令则再按一次【STEP】 按键。

3.4.2 运行模式

在运行模式中可以点击左下角的【运行次数】按钮来设置程序的运行次数。默认为运行一次。点击弹出框中的【循环运行】按钮可以使程序无限循环运行。

运行模式时程序上方显示已运行次数与总设置运行次数，格式为“已运行次数/共设置运行次数”。

运行过程中，可以修改运行次数，修改后机器人在运行设置的次数后停止。例如原设置运行 200 次，已运行 156 次，此时设置运行次数为 3 次，则机器人在继续运行三次后停止。

运行速度=指令速度*上方状态栏的速度比率。

3.4.3 远程模式

远程模式支持两种外接设备，数字 IO 和 Modbus 触摸屏。

设备优先级为：Modbus>数字 IO，当两个外接设备都在连接时，可通过 Modbus 触摸屏来控制数字 IO 的使能。

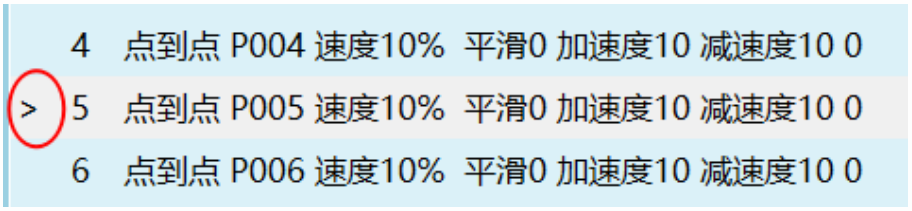
当示教器被拔下后，触发远程 IO 信号，将自动进入远程模式。

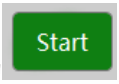
3.4.4 从当前行运行

在示教模式下打开作业文件，选中某一行，点击操作 

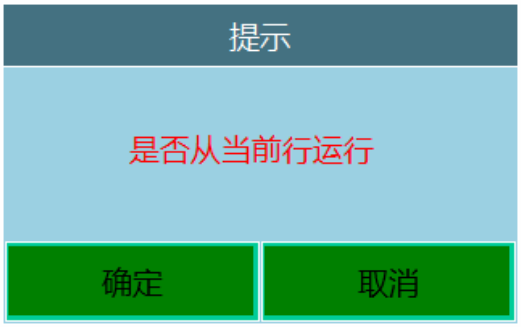
按钮，点击从此运行

，作业文件会出现>符号；



切至运行模式，点击启动 

运行会提示弹窗。



点击确认按钮则从选中行运行，点击取消则从首行开始运行。

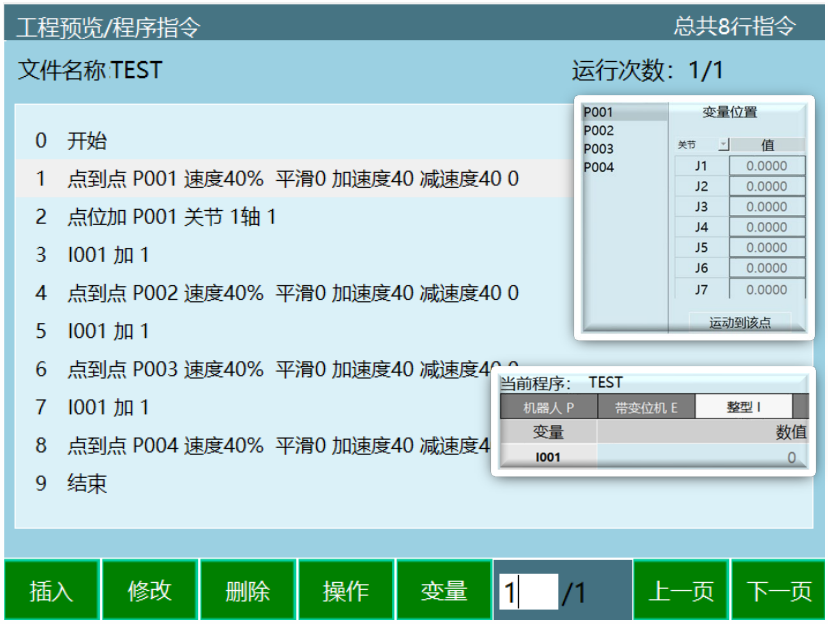
3.4.5 断点运行

3.4.5.1 运行模式断点

运行过程中（第一条指令除外），切换至其他模式时导致运行中断，会将中断时的变量状态、程序运行位置存为断点，再次运行时，会弹出提示框询问“是否继续运行当前程序”，选择“确定”则从断点处继续运行，选择“取消”则断点消失从第一条指令重新运行。

- **断点状态查看**：断点后切换到示教模式后，可以通过上电查看断点时的位置/数值变量状态。

例：P001 与 I001 初始状态如图，运行过程中发生改变 P001 J1+1、I001+1。



运行到第 6 行时 P001 J1=1、I001=2，切换示教模式产生断点，切换到示教模式后查看 P001、I001 显示为初始值，此时按【DEADMAN】上电，显示为 P001 J1=1、I001=2，下电恢复初始值。

- **断点解除：**产生断点后，进行回零、复位、单步运行指令、运行其他程序、运行到该点、弹框中选择“取消”、插入/删除/移动/剪切/复制指令、修改局部数值/局部位置变量、重启控制器、修改机器人参数等操作会解除断点，再运行程序会从第一条指令开始。

3.4.5.2 示教模式

示教模式也存在“断点”，单步程序过程中如果有改变局部变量的指令，下电后再上电可以查看“断点”时的局部变量值。

进行回零、复位、单步指令中下电、运行其他程序、运行到该点、修改局部数值/局部位置变量并单步指令、重启控制器、修改机器人参数等操作会解除“断点”

3.4.5.3 远程模式断点

使用 io 预约程序默认执行断点，如不需远程断点，在设置-操作参数-远程模式是否使用断点执行中关闭

3.4.6 提前执行功能

运动指令时间参数设置时生效，参数点位 ms。

上图示例程序中，MOVJ 指令后插入 DOUT 指令；MOVJ 指令 TIME 参数填 1000ms，则运行时会提前 1s 执行下一条指令，例如 MOVJ 指令会执行 3s，则 MOVJ 指令运行 2 秒执行 DOUT 并继续执行 MOVJ 到 P001。

坐标类	~~切换工具手~~			
	~~切换工具坐标~~		打磨	棱边打磨
	~~用户坐标转换~~			继续打磨
	切换外部轴			结束打磨
网络通讯类	发送数据		电批	~~电批~~
	解析数据			
	读取			
	打开数据			
	关闭数据			
	输出信息			
	获取信息连接状态			
位置变量类	~~用户坐标修改~~			
	~~工具坐标修改~~			
	读取点位			
	点位加			
	点位减			
	点位该			
	复制点位			
	点位全加			
	点位全减			
	点位全改			
	轨迹偏移			
程序控制器	开启线程			
	退出线程			
	暂停运行			
	继续运行			
	停止运行			
	重新运行			

3.5 机器人运动速度

本系统示教模式速度、运行模式速度、远程模式速度需分开设置

3.5.1 示教模式速度

关节点动速度=关节轴最大点动速度*示教速度（最大速度限制=关节轴最大点动速度*50%）；

直角点动速度=直角坐标最大点动速度*示教速度（最大速度限制=300mm/s）；

回零速度=额定速度*示教速度*10%（最大速度限制=额定速度*10%）；

关节回安全点速度=额定速度*示教速度*10%（最大速度限制=额定速度*10%）；

直线回安全点速度=100mm/s*示教速度（最大速度限制=100mm/s）；

运动到该点速度=示教速度*示教速度;

单步关节速度=额定速度*示教速度*指令速度（最大速度限制=额定速度*30%）；

单步直角速度=示教速度*指令速度（最大速度限制=300mm/s）；

3.5.2 运行模式速度

运行点到点速度=额定速度*运行速度*指令速度

运行直线速度=运行速度*指令速度

3.5.3 远程模式速度

远程点到点速度=额定速度*远程速度*指令速度

远程直线速度=远程速度*指令速度

3.5.4 远程 IO 速度修改方式

注：远程模式示教盒禁止修改速度操作，需在示教模式提前设置，远程速度默认 15%

1 进入设置-机器人参数-运动参数界面。

2 点击修改，修改远程模式速度，点击保存，切至运行模式查看。

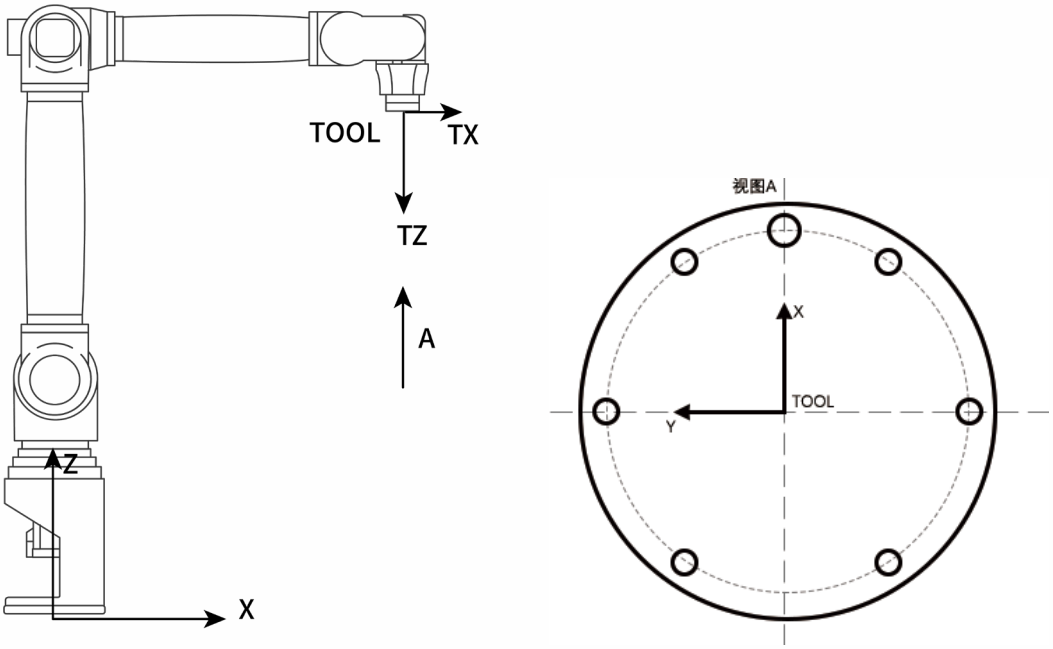
3 修改成功。

第4章 工具手与用户坐标

4.1 工具手标定

4.1.1 工具坐标系

法兰盘中心：默认工具坐标系的原点，法兰盘中心指向法兰盘定位孔方向为+X 方向，垂直法兰向外为+Z 方向，最后根据右手法则即可判定 Y 方向。新的工具坐标系都是相对默认的工具坐标系变化得的。



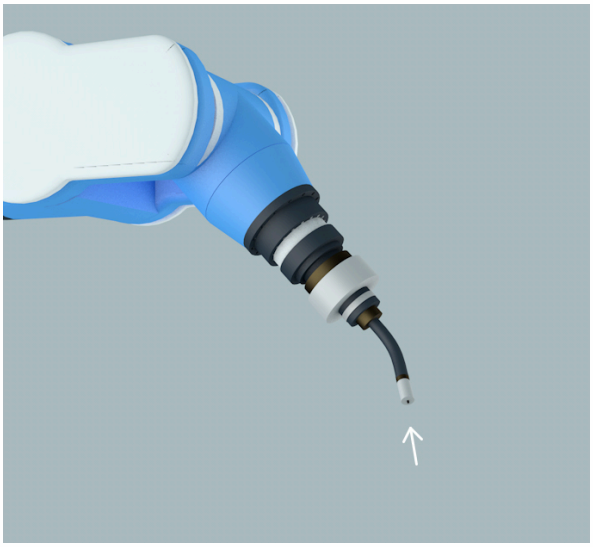
TCP:TOOL CENTER POINT,即工具中心点

机器人轨迹及速度：指 TCP 点的轨迹和速度。

TCP 一般设置在手爪中心，焊丝端部，点焊静臂前端等等。

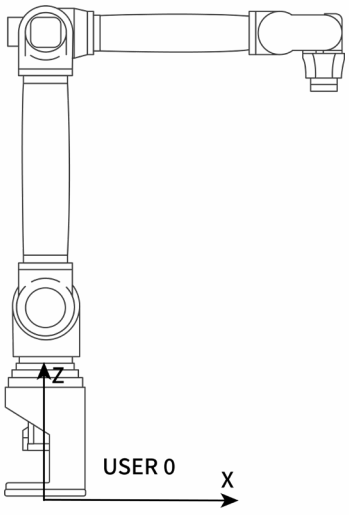
为了描述一个物体在空间的位置，需在物体上固定一个坐标系，然后确定该坐标系位姿（原点位置和三个坐标轴姿态），即需要 7 个 DOF 来完整描述该刚体的位姿[1]。对于工业机器人，需要在末端法兰盘安装工具（Tool）来进行作业。为了确定该工具（Tool）的位姿，在 Tool 上绑定一个工具坐标 TCS（Tool Coordinate System），TCS 的原点就是 TCP（Tool Center Point，工具中心点）。在机器人轨迹编程时，需要将 TCS 在其他坐标系的位姿记录到程序中执行。

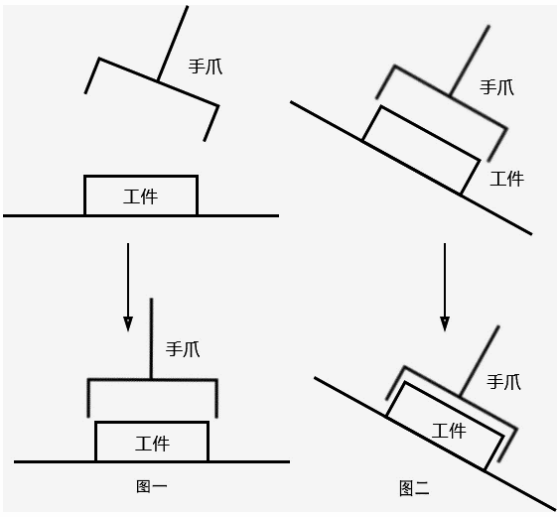
工业机器人一般都事先定义了一个 TCS，TCS 的 XY 平面绑定在机器人第六轴的法兰盘平面上，TCS 的原点与法兰盘中心重合。显然 TCP 在法兰盘中心。ABB 机器人把 TCP 称为 tool0，REIS 机器人称之为 _tnull。虽然可以直接使用默认的 TCP，但是在实际使用时，比如焊接，用户通常把 TCP 点定义到焊丝的尖端（实际上是焊枪 tool 的坐标系在 tool0 坐标系的位姿），那么程序里记录的位置便是焊丝尖端的位置，记录的姿态便是焊枪围绕焊丝尖端转动的姿态。



思考

从思考 1 中，我们知道工具坐标系是运动中的一个研究对象，但是它在实际调试过程中，又起到了什么作用呢？思考下图一、图二的手爪姿态和位置是如何调整得到的？





推测：根据思考可以得出两个推测：

推测 1：若图 1 中的手爪有一个旋转点，使手爪直接绕着这个旋转点选择就可以。

推测 2：若图二中有一个手爪的前进方向就可以直接移动过去了。

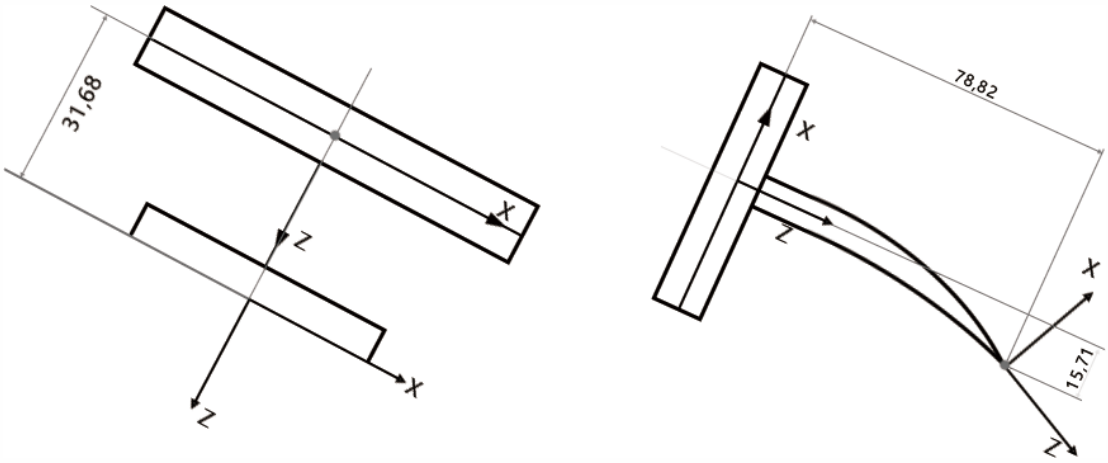
结论：建立工具坐标系的作用:

1、 确立工具的 TCP 点（即工具中心点），方便调整工具状态。

2、 确定工具进给方向，方便工具位置调整。

4.1.2 工具坐标系特点

新的工具坐标系是相对于默认的工具坐标系变化得到的，新的工具坐标系的位置和方向始终同法兰盘保持绝对的位置和姿态关系，但在空间上是一直变化的。



4.1.3 工具手参数设置

点击设置里的【工具手标定】就能进入工具手标定界面，如图

若有工具的详细参数，在该界面下，用户可以直接填写工具末端偏移的相关参数，不需进行七点标定。

进入该界面时会自动读取控制器中已保存的工具手尺寸参数（默认各项为 0），若更换工具手请重新填写。

详细参数设置步骤如下:

1. 打开工具手标定界面，下面表格是对每个参数的介绍:

参数	作用
X 轴方向偏移	工具末端相对于法兰中心，沿笛卡尔坐标系 X 轴方向的偏移长度（毫米）。
Y 轴方向偏移	工具末端相对于法兰中心，沿笛卡尔坐标系 Y 轴方向的偏移长度（毫米）。
Z 轴方向偏移	工具末端相对于法兰中心，沿笛卡尔坐标系 Z 轴方向的偏移长度（毫米）。
绕 A 轴偏移	工具末端相对于法兰中心，绕笛卡尔坐标系 X 轴方向的偏移角度（°）
绕 B 轴偏移	工具末端相对于法兰中心，绕笛卡尔坐标系 Y 轴方向的偏移角度（°）
绕 C 轴偏移	工具末端相对于法兰中心，绕笛卡尔坐标系 Z 轴方向的偏移角度（°）

2. 点击【修改】按钮。
3. 填写工具对应的各项参数，其中各参数作用如上表所示;
4. 确认无误后点击【保存】按钮，设置成功。

进行数据量取前请将法兰盘平行于水平面。

点击【清除】按钮可以将已填写的参数清零。

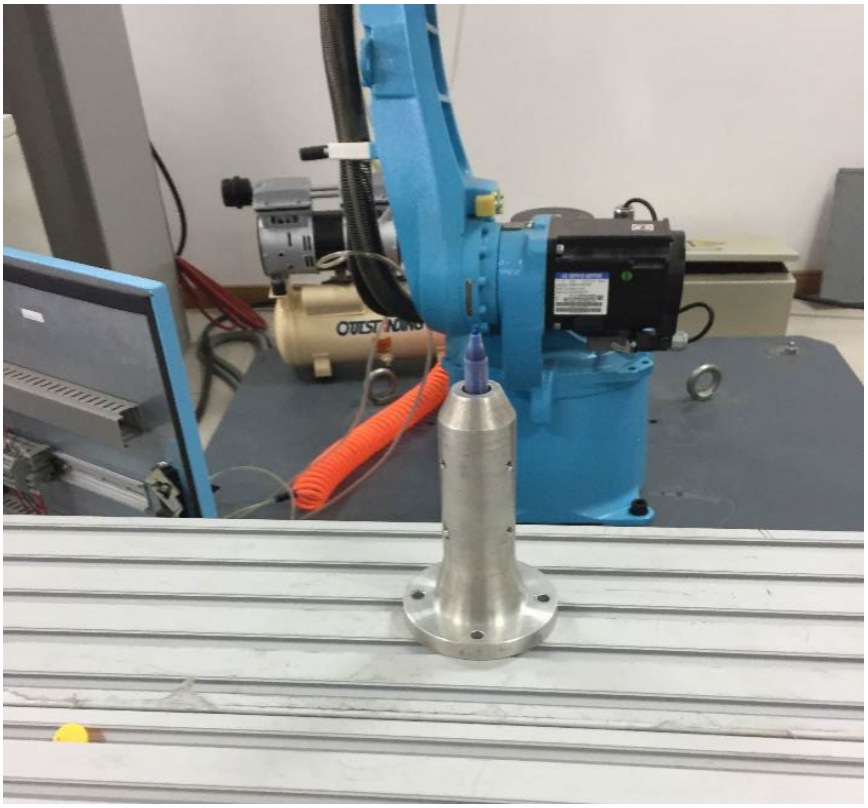
若在参数设置过程中点击底部操作区的【返回】或者【七点标定】按钮，则跳转到相应界面，未保存的设置参数不会保留。

4.1.4 7 点标定

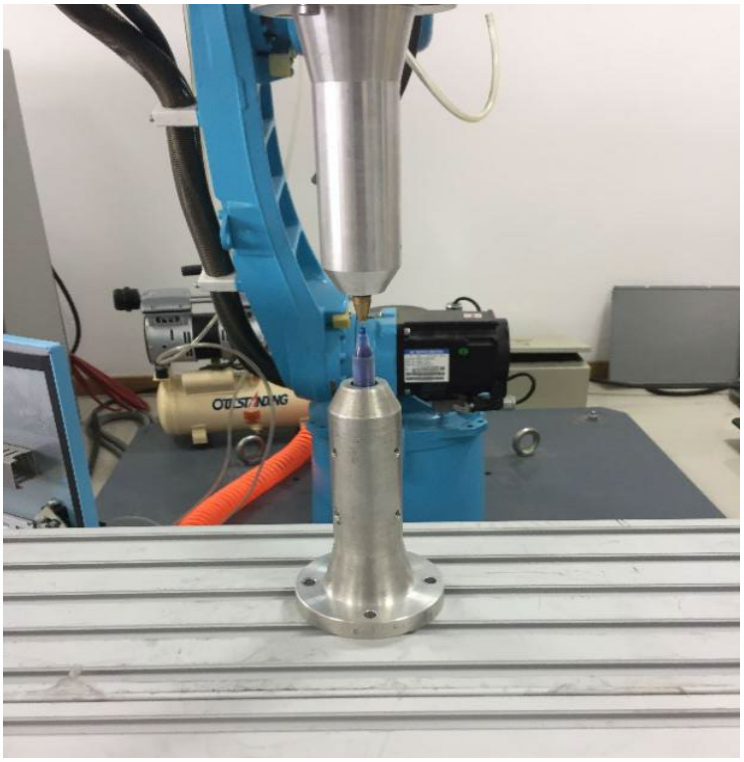
点击底部的【七点标定】按钮进入七点标定界面，如图

若没有工具的详细参数，可以进行 TCP 标定，自动计算出工具各项尺寸参数。具体标定步骤如下：

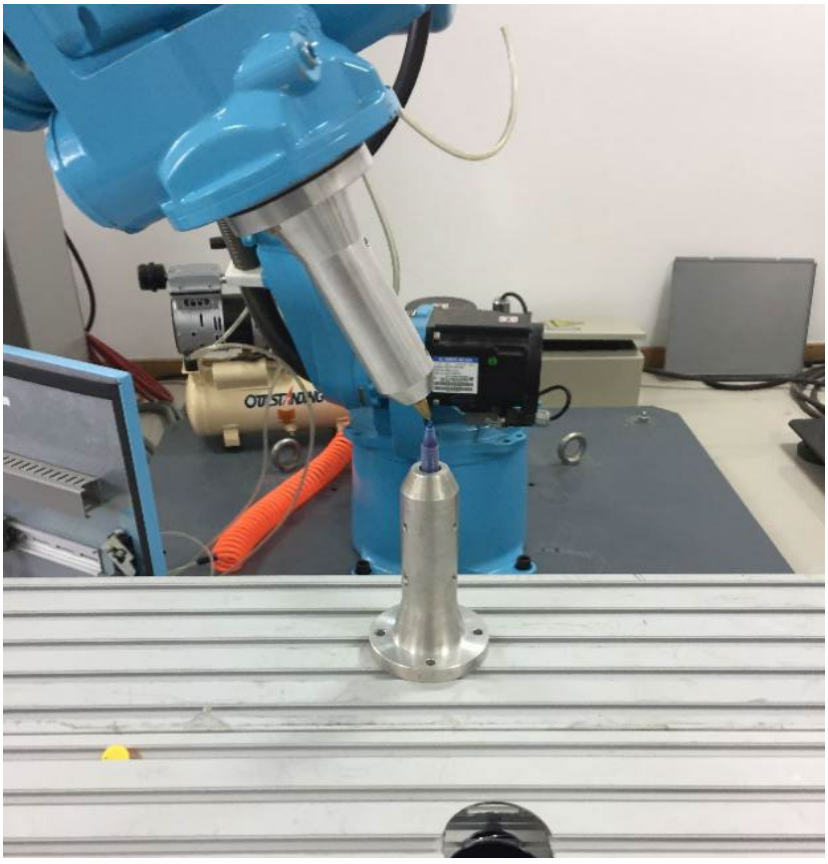
1. 现在以笔尖为参考点，并确保此参考点固定，如下图所示。



2. 将工具末端垂直且正对参考点，点击界面“TC1”所对应的【标定】按钮，如下图所示。

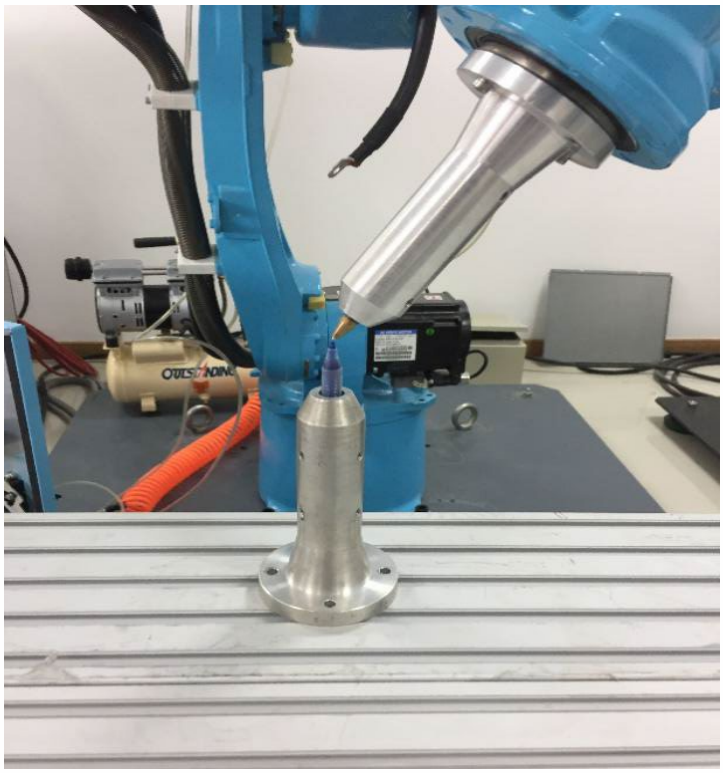


3. TC2 标定：将机器人切换一个姿势，末端正对参考点，点击该行所对应的【标定】按钮，如下图所示。

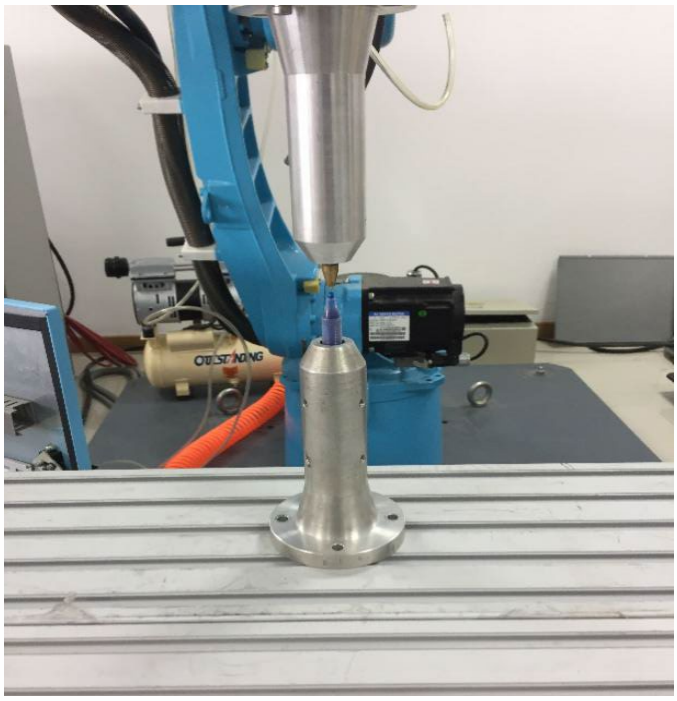


4. TC3 标定：将机器人切换一个姿势，末端正对参考点，点击该行所对应的【标定】按钮，如下图所示。

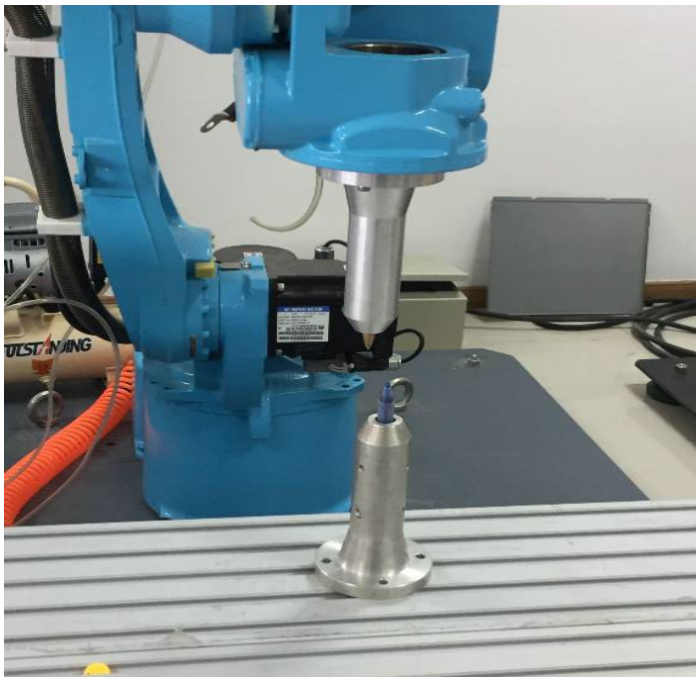
5. TC4 标定：将机器人切换一个姿势，末端正对参考点，点击该行所对应的【标定】按钮，如下图所示。



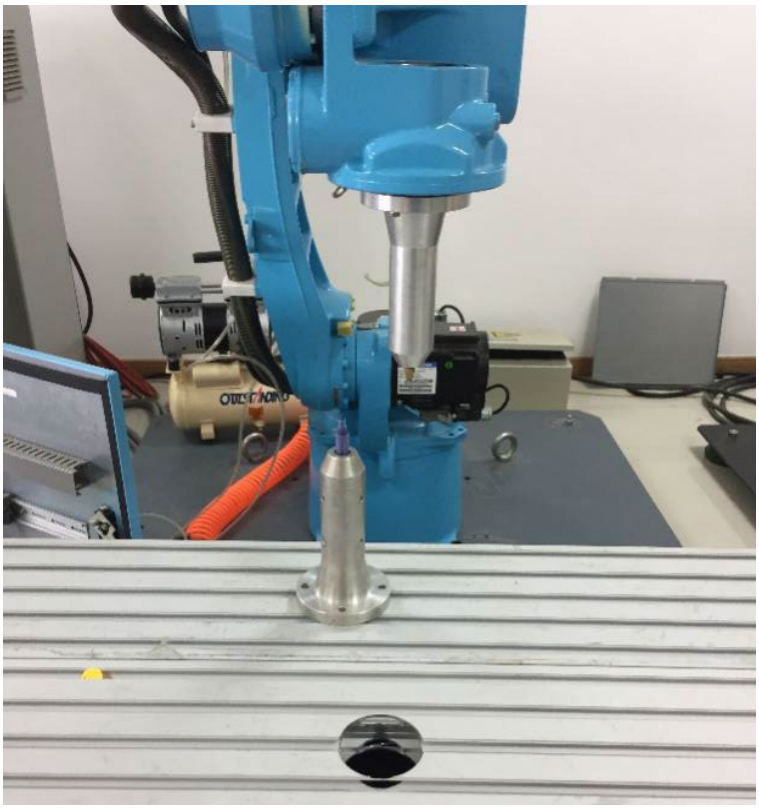
6. TC5 标定：将工具末端垂直且正对参考点（同 TC1），点击该行所对应的【标定】按钮，如下图所示。



7. TC6 标定：在 TC5 的基础上，沿笛卡尔坐标系 X 轴负方向移动任意距离，点击该行所对应的【标定】按钮，如下图所示。



8. TC7 标定：在 TC6 的基础上，沿笛卡尔坐标系 Y 轴正方向移动任意距离，点击该行所对应的【标定】按钮，如下图所示。



9. 点击 【运行至该点】，可以查看标定是否准确；

10. 点击【计算】按钮，标定成功。

若在标定过程中对某点标定后不满意，可以点击该行所对应的【取消标定】按钮，取消标定后再次标定该点。

点击底部的【演示】按钮可以打开“演示”界面，讲解如何进行工具标定。

点击底部的【返回】按钮可以返回“工具手标定”界面。

4.1.5 12/15 点标定

12 点/15 点/20 点标定公用一个标定界面，标定前 15 个点即为使用 15 点标定法。

12 点标定即 15 点标定不标最后三个点（13-15），标定结果只有工具手的 XYZ 轴方向偏移，无绕 ABC 旋转的数值。

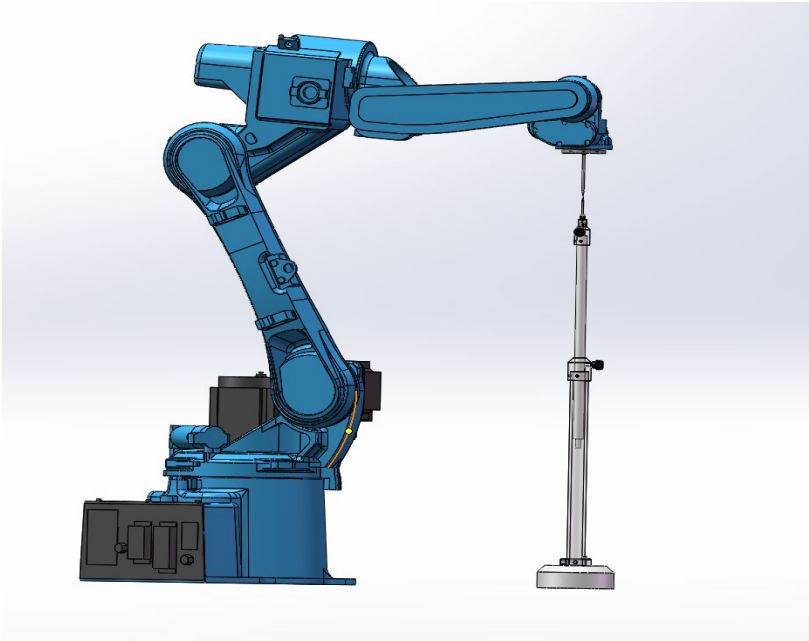
点击“工具手标定”界面底部的【20 点标定】按钮，进入标定界面，如图。

具体标定步骤如下：

1. 找到一个参考点（标定锥尖端为参考点），并确保此参考点固定。
2. 开始插入位置点，每插入一点，点击【标记该点】，插入十五个点

   具体步骤如下：

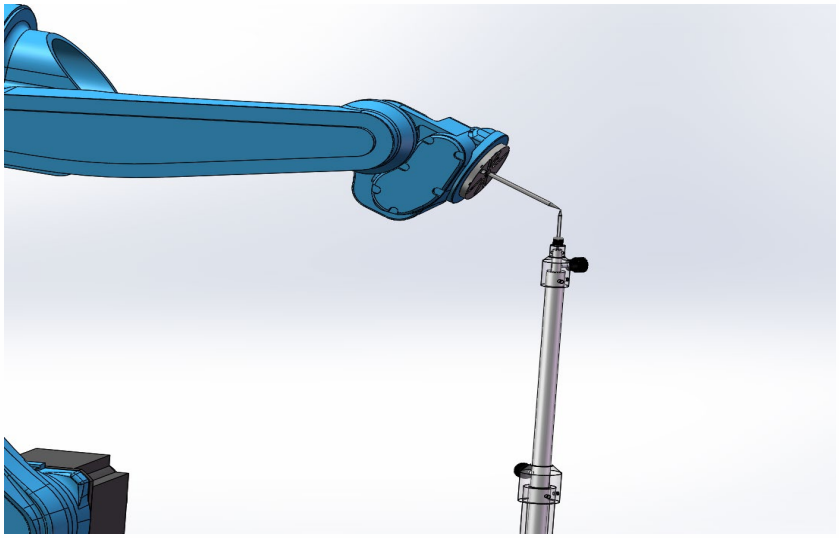
   1) 第一个点　机器人回归零点，通过直角坐标将机器人尖端对准标定锥尖端，标定第一个点；



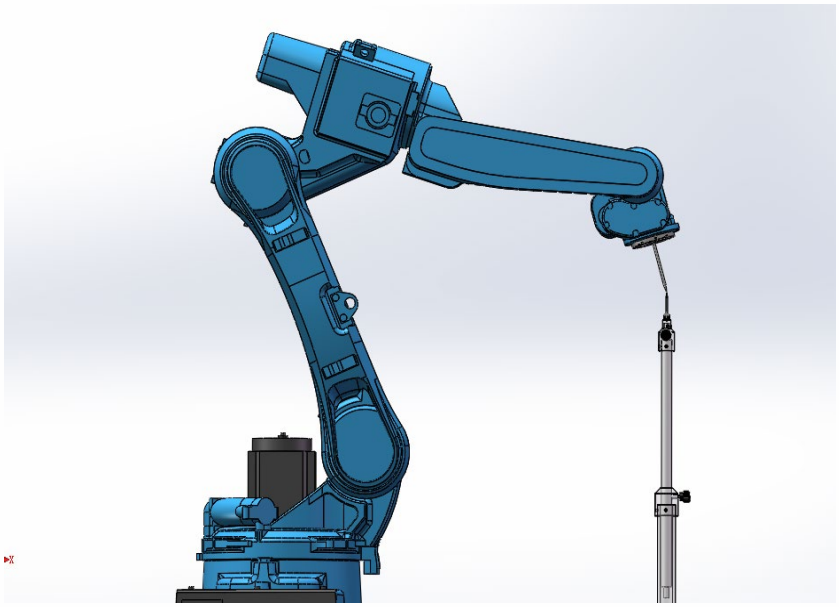
   2) 第二个点　在第一个点的基础上，通过直角坐标系将 C 旋转 180 度；尖端对齐标定第二点；

3) 第三个点　机器人回归零点，通过直角坐标系将机器人尖端对准标定锥 尖端；标定第三个；（**与第一个点相同**）

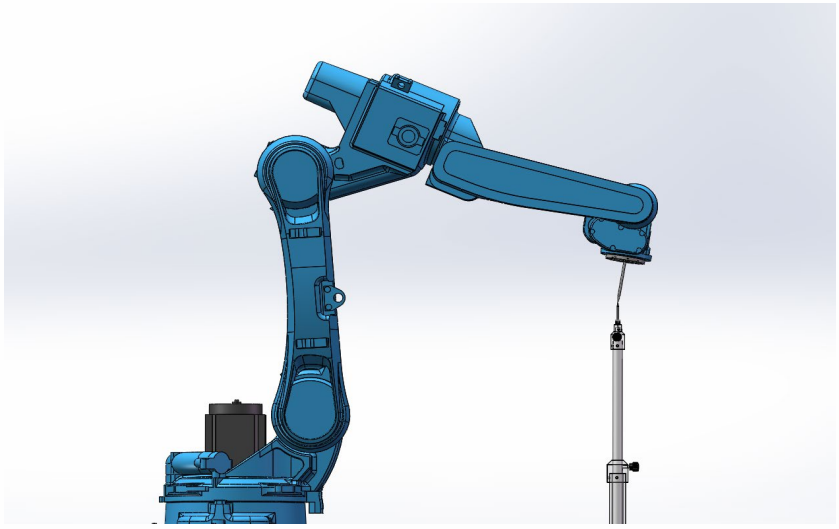
4) 第四个点　在第三个点的基础上，通过直角坐标系做 B-，度数位于 30°-60°，尖端对齐标定第四个点；



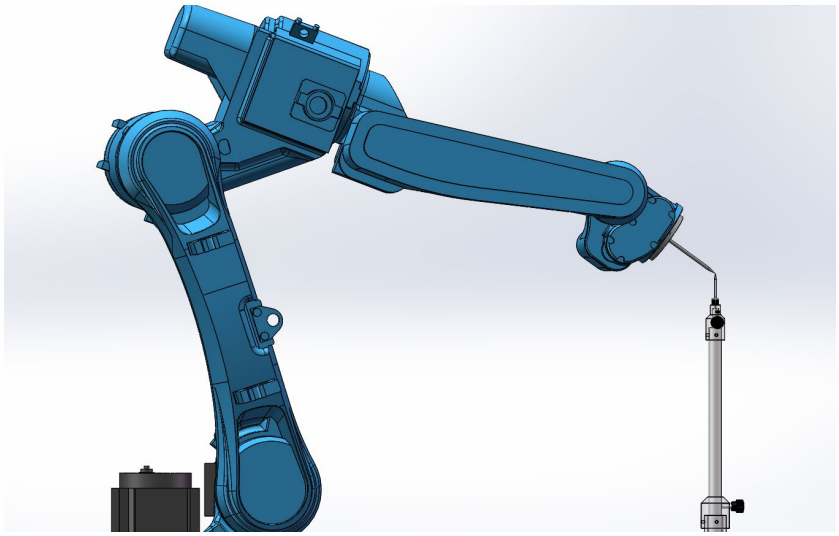
5) 第五个点　在第四个点的基础上，通过直角坐标系做 B+，J5>-90°,将机器人尖端对准标定锥 尖端，标定第五个点；



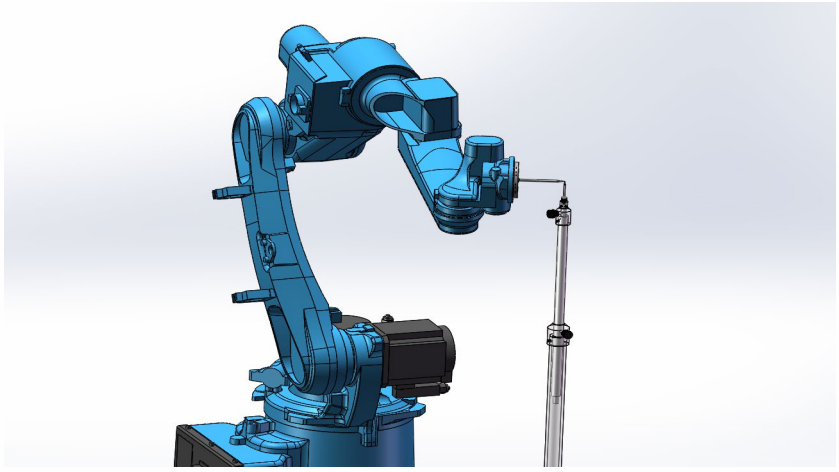
6) 第六个点　选中第一个点，并将机器人移动到第一个点，在第一个点的基础上，通过直角坐标系做 B+，J5>-90°,尖端对齐标定第六个点；



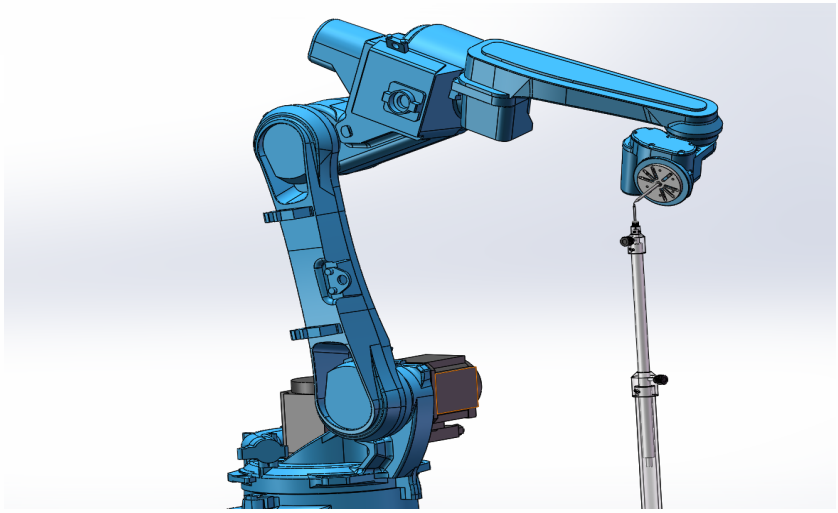
7) 第七个点 在第一个点的基础上，通过直角坐标系做 B-，J5>-90°,尖端对齐标定第七个点；



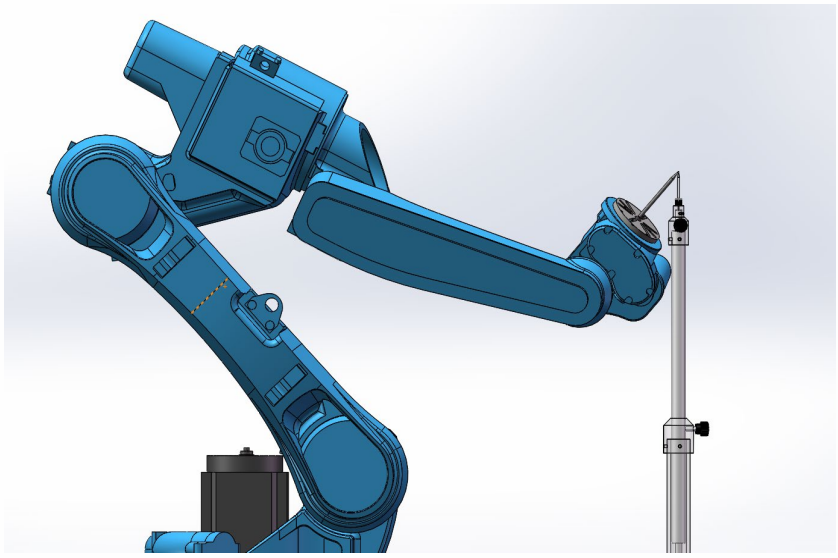
8) 第八个点 在第七个点的基础上，通过直角坐标系做 A+，旋转 90°，J5>-90°,尖端对齐标定第八个点；



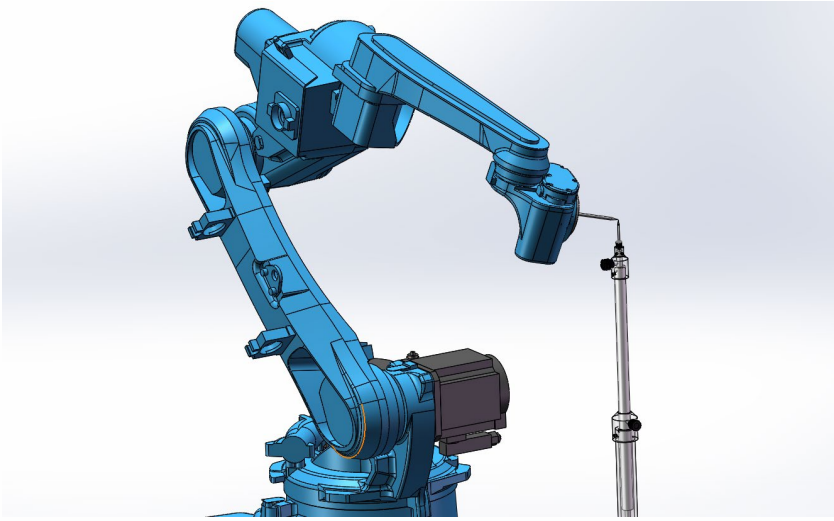
9) 第九个点　　在第七个点的基础上，通过直角坐标系做 A-，旋转 90°，J5>-90°,尖端对齐标定第九个点；



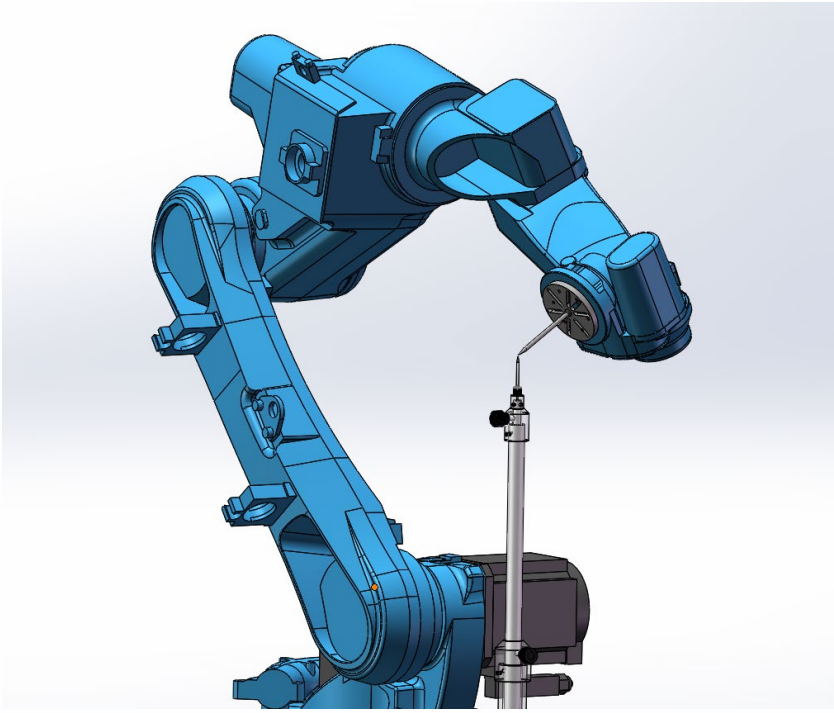
10) 第十个点　　机器人回到第一个点，通过关节坐标系点动五轴，使五轴向上，J5<-90°,将尖端对齐，标定第十个点；



11) 第十一点　　机器人在第十点的基础上，通过直角坐标系做 A+，旋转 90°，J5<-90°,尖端对齐标定第十一个点；



12) 第十二点　机器人在第十点的基础上，通过直角坐标系做 A-，旋转 90°，J5<-90°,尖端对齐标定第十一个点；



13) 第十三点　机器人回到零点位置，调整机器人姿态，使机器人末端工具尖端竖直朝下，将标定尖端与标定锥 对齐，标定第十三个点；

14) 第十四点　在第十三点的基础上，通过直角坐标系做 X-，机器人位移一段距离，直接点击标定第十四点

15) 第十五点　在第十四点的基础上，通过直角坐标系做 Y+，机器人位移一段距离，直接点击标定第十五点；

3. 完成标记后，点击【计算】。

【取消标定】：若在标定过程中对某点标定后不满意，可以点击该行所对应的【取消标定】按钮，取消标定后再次标定该点。

【运行到该点】：每标定完一个点可以点击【运行到该点】，则机器人会运行到该点。

【将结果位置标为零点】：将标定补偿后的位置设置为当前机器人的零点位置。

【清除所有标定点】：标定点位会保存到控制器中，只有点击取消标定、清除所有标定点以及切换工具手进标定界面后，标定结果才会清除

注意

各点的姿势，请尽量取任意方向的姿势。取的姿势朝一定方向旋转的话，有些时候精度不准确。

标定过程中请保持参考点固定，否则标定误差增大。

点击底部的【返回】按钮，可以返回“工具手标定”界面。

4.1.6 20 点标定

12 点/15 点/20 点标定公用一个标定界面，标定所有 20 个点即为使用 20 点标定法。

点击“工具手标定”界面底部的【二十点标定】按钮，进入“二十点标定”界面，如图。

具体标定步骤如下：

1. 找到一个参考点（笔尖为参考点），并确保此参考点固定。
2. 开始插入位置点，每插入一点，点击【标记该点】，插入 20 个点，每个点的姿态差异越大越好。

   厂家建议：标定步骤，第一点工具手姿态垂直向下，第二点走 A+轴，第三点走 A+，第四点走 A+，第五点走 A-，第六点走 A-，第七点走 A-，第八点走 B+，第九点走 B+，第十点走 B+，第十一点走 B-，第十二点走 B-，第十三点走 B-，其余点主要动 C 轴成米字形排布标定
3. 完成 20 点标记后，点击【计算】。

   【取消标定】：若在标定过程中对某点标定后不满意，可以点击该行所对应的【取消标定】按钮，取消标定后再次标定该点。

   【运行到该点】：每标定完一个点可以点击【运行到该点】，则机器人会运行到该点。

   【将结果位置标为零点】：将标定补偿后的位置设置为当前机器人的零点位置。

   【清除所有标定点】：标定点位会保存到控制器中，只有点击取消标定、清除所有标定点以及切换工具手进标定界面后，标定结果才会清除

注意

各点的姿势，请尽量取任意方向的姿势。取的姿势朝一定方向旋转的话，有些时候精度不准确。

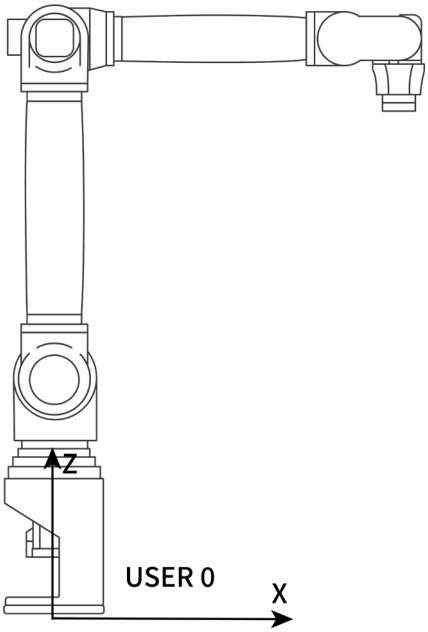
标定过程中请保持参考点固定，否则标定误差增大。

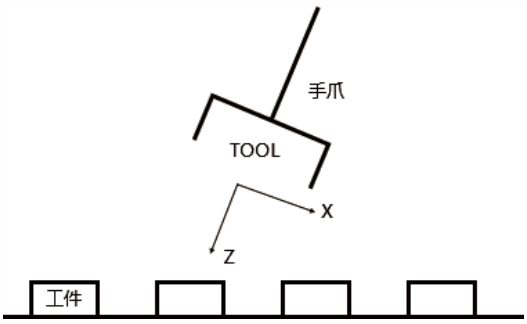
4.2 用户坐标系

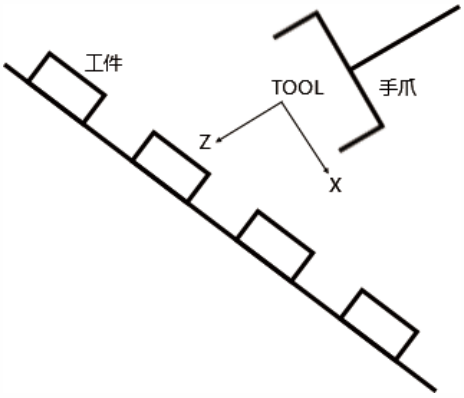
4.2.1 用户坐标系作用

定义：默认的用户坐标系：默认的用户坐标系 User0 和直角坐标系重合。新的用户坐标系都是基于默认的用户坐标系变化得到的。

思考：从思考 1 中我们知道用户坐标系是运动中的一个参考对象，但是它在实际调试过程中，又起到了什么作用呢？







推测：从图中可以看出，如果使用默认的用户坐标系 User 0 或者直角坐标系将很难对每个工件位置进行调试，但如果存在某个坐标系的两个方向正好平行于工作台面的话，那就方便多了。

结论：用户坐标系作用

1. 确定参考坐标系；
2. 确定工作台上的运动方向，方便调试。

用户坐标系特点

新的用户坐标系是根据默认的用户坐标系 User 0 变化得到的，新的用户坐标系的位置和姿态相对空间是不变化的。

4.2.2 用户坐标参数设置

点击“设置”界面的【用户坐标标定】按钮，进入“用户坐标”界面，如图。

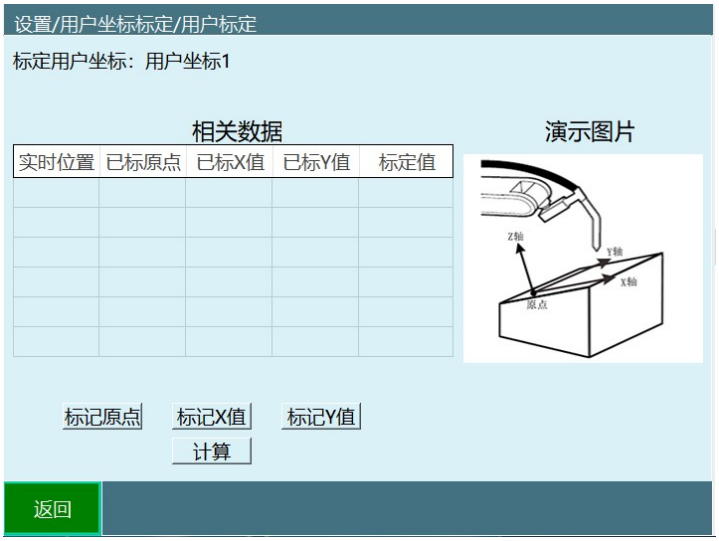
用户坐标的参数如下

参数	作用
X 值	用户坐标原点相对机器人基座原点 X 轴方向的偏移
Y 值	用户坐标原点相对机器人基座原点 Y 轴方向的偏移
Z 值	用户坐标原点相对机器人基座原点 Z 轴方向的偏移
A 值	用户坐标系相对直角坐标系绕 X 轴方向的旋转角（弧度）
B 值	用户坐标系相对直角坐标系绕 Y 轴方向的旋转角（弧度）
C 值	用户坐标系相对直角坐标系绕 Z 轴方向的旋转角（弧度）

若有精确数值请直接填写，注意 ABC 三个值为弧度。

4.2.3 用户坐标系标定

点击“用户坐标标定”界面底部的【用户标定】按钮，进入“用户标定”界面，如图。



用户坐标系的标定请遵循以下步骤：

1. 将机器人末梢移动到期望为用户坐标系原点的位置，点击“标定原点”按钮；
2. 将机器人相对于用户坐标系原点向期望为用户坐标系 X 轴正方向的位置移动任意距离，点击“标定 X 轴”按钮；
3. 将机器人相对于用户坐标系原点向期望为用户坐标系 Y 轴正方向的位置移动任意距离，点击“标定 Y 轴”按钮。

注意
用户坐标系的 Y 轴若没有标定准确，系统会自动补偿。

点击界面底部【返回】按钮，返回用户坐标标定界面。

第5章 数值变量

本章主要说明本控制系统的变量的相关情况。

5.1 变量的名称

	类型	数量	示例
全局数值变量	全局整形型、GINT	990 个	GI001
	全局实数型、GDOUBLE		GD001
	全局布尔型、GBOOL		GB001
局部数值变量	局部整形型、INT	每个作业文件 999 个	I001
	局部实数型、DOUBLE		D001
	局部布尔型、BOOL		B001

5.2 变量类指令

- **INT**

定义一个局部 INT 变量并赋值，需将指令插在程序头部。

INT	功能	定义局部 INT 变量并赋值	
	参数	变量名	0-999
		变量值来源	常量 INT DOUBLE BOOL GINT GDOUBLE GBOOL
		新参数	常量
		来源参数	已有变量名
	使用示例	INT I001=1	

- **DOUBLE**

定义一个局部 DOUBLE 变量并赋值，需将指令插在程序头部。

DOUBLE	功能	定义局部 DOUBLE 变量并赋值	
	参数	变量名	0-999
		变量值来源	常量 INT DOUBLE BOOL GINT GDOUBLE GBOOL
		新参数	常量
		来源参数	已有变量名
	使用示例	DOUBLE D001=1	

- **BOOL**

定义一个局部 BOOL 变量并赋值，需将指令插在程序头部。

BOOL	功能	定义局部 BOOL 变量并赋值	
	参数	变量名	0-999
		变量值来源	常量 INT DOUBLE BOOL GINT GDOUBLE GBOOL
		新参数	常量
		来源参数	已有变量名
	使用示例	BOOL A001=1	

- **SETINT**

给 INT 变量赋值。

SETINT	功能	给 INT 变量赋值	
	参数	变量	INT GINT
		变量值来源	常量 INT DOUBLE BOOL GINT GDOUBLE GBOOL
		新参数	常量
		来源参数	已有变量名
	使用示例	SETINT I001=1	

- **SETDOUBLE**

给 DOUBLE 变量赋值。

SETDOUBLE	功能	给 DOUBLE 变量赋值	
	参数	变量	INT GINT
		变量值来源	常量 INT DOUBLE BOOL GINT GDOUBLE GBOOL
		新参数	常量
		来源参数	已有变量名
	使用示例	SETDOUBLE D001=1	

- **SETBOOL**

给 BOOL 变量赋值。

SETBOOL	功能	给 BOOL 变量赋值	
	参数	变量	INT GINT
		变量值来源	常量 INT DOUBLE BOOL GINT GDOUBLE GBOOL
		新参数	常量
		来源参数	已有变量名
	使用示例	SETBOOL A001=1	

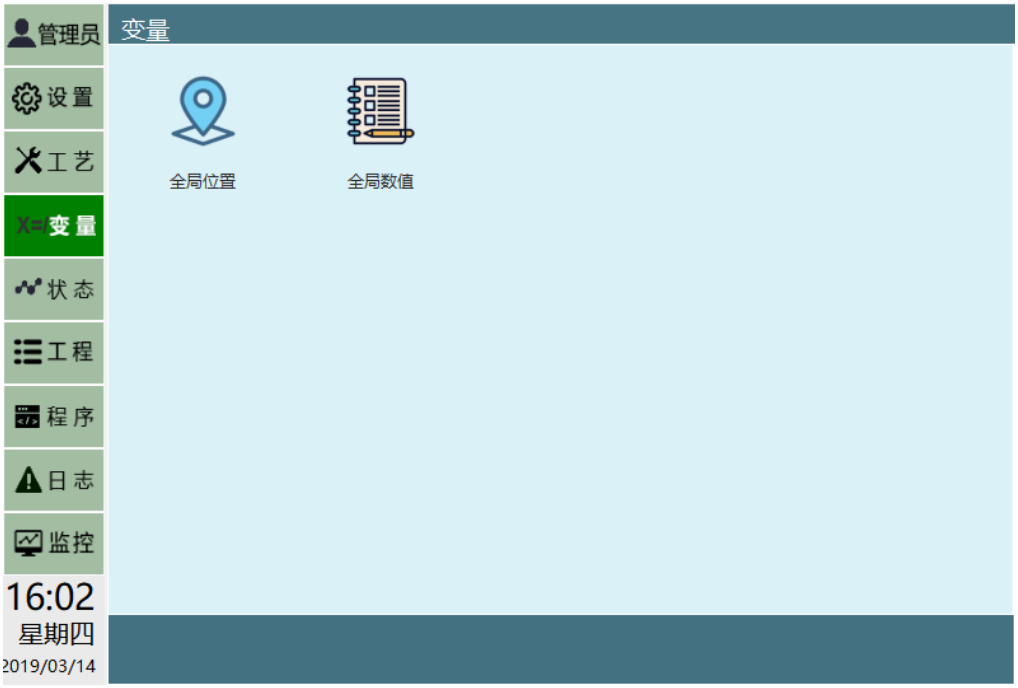
- **FORCESET**

在程序运行过程中，将当前缓存中的全局变量值写入变量文件中。

FORCESET	功能	在程序运行过程中，将当前缓存中的全局变量值写入变量文件中。	
	参数	变量类型	GINT GDOUBLE GBOOL
		变量名	变量名
	使用示例	FORCESET GI001	

5.3 全局数值变量

全局数值变量是可以作用于所有机器人所有程序的变量，如机器人 1 的程序 AA 和机器人 2 的程序 BB 可以同时使用同一个全局数值变量。本节将主要说明全局变量界面的使用，以及位置、数值变量的使用方法。



试想机器人完成一道工序需要那么多的指令，如果我们每次插入指令，设置变量，这是多么繁琐的工作，基于此我们加入了数值变量以便调用。例如“WHILE（INT001=10）...END（WHILE）”这样的指令，在机器人完成某道工序的程序中很多，我们直接调用你预先设置好的数值变量。

同时全局数值变量可以用来在主程序、被调用的子程序以及后台程序之间传递信息，用以做逻辑判断使用。

数值型变量储存的是数值，包含了整数型变量、实数型变量、布尔型变量三种。

注：全局变量赋值后会直接保存到参数

5.3.1 全局布尔型变量

全局布尔型变量保存的是字节，在该界面中可以修改每一个变量的数值、注释。各参数的意义如下：

- 变量名即该变量的编号，全局布尔型变量的名字为 GBxxx。
- 数值即该变量的值，布尔型变量的值的范围为“0/1”。
- 注释为用户给该变量定义的注释，方便用户标记该变量的作用，范围为任意值，可为中文。

5.3.2 全局整数型变量

全局整数型变量保存的是整数，在该界面中可以修改每一个变量的数值、注释。各参数的意义如下：

- 变量名即该变量的编号，全局整数型变量的名字为 GIxxx。
- 数值即该变量的值，整数型变量的范围为整数。
- 注释为用户给该变量定义的注释，方便用户标记该变量的作用，范围为任意值，可为中文。

5.3.3 全局浮点型变量

全局实数型变量保存的为实数，在该界面中可以修改每一个变量的数值、内容、注释。各参数的意义如下:

- 变量名即该变量的编号，全局实数型变量的名字为 GDxxx。
- 数值即该变量的值，浮点型变量的范围为实数。
- 注释为用户给该变量定义的注释，方便用户标记该变量的作用，范围为任意值，可为中文。

点击要修改的数据类型，再选择变量名，点击【修改】，则可以修改数值、注释。而后点击【保存】。点击【清除】则可以清除你所选择的数据。

5.3.4 全局数值变量使用

5.3.4.1 定义全局数值变量

在使用变量之前请定义变量，定义变量的方法如下:

1. 点击左侧“变量”按钮，进入变量界面;
2. 点击全局数值变量;

3. 选择对应的变量编号，点击“修改”按钮；

4. 在数值与注释处填写需要的值；

5. 未手动定义过得变量，默认为 0。

5.3.4.2 通过运算类指令为全局数值变量赋值

通过 ADD、SUB、MUL、DIV、MOD 指令可以对全局变量进行计算。

注意：全局布尔变量不可进行计算！

5.3.4.2.1 ADD

加法运算（+）。

公式：变量类型（变量名）=变量类型（变量名）+变量值来源（参数）

若要对全局整数或全局数值变量进行计算，则在变量类型处选择 GINT 或 GDOUBLE。

其中变量值来源若选择自定义，则参数则可以手动填写在“新参数”处。同时也可以为其它变量值。

例 1
前提：GI001=1
指令：ADD GI001 1
意义：GI001=GI001+1
结果：GI001=2
例 2
前提：GI001=1 GI002=2
指令：ADD GI001 GI002
意义：GI001=GI001+GI002
结果：GI001=3

5.3.4.2.2 SUB

减法运算（-）

公式：变量类型（变量名）=变量类型（变量名）-变量值来源（参数）

若要对全局整数或全局数值变量进行计算，则在变量类型处选择 GINT 或 GDOUBLE。

其中变量值来源若选择自定义，则参数则可以手动填写在“新参数”处。同时也可以为其它变量值。

例
前提：GD001=3.4
指令：SUB GD001 1.1
意义：GD001=GD001-1.1
结果：GD001=2.3

5.3.4.2.3 MUL

乘法运算（*）

公式：变量类型（变量名）=变量类型（变量名）*变量值来源（参数）

若要对全局整数或全局数值变量进行计算，则在变量类型处选择 GINT 或 GDOUBLE。

其中变量值来源若选择自定义，则参数则可以手动填写在“新参数”处。同时也可以为其它变量值。

例
前提：GD001=3.4 GI001=2
指令：MUL GD001 GI001
意义：GD001=GD001*GI001
结果：GD001=6.8

5.3.4.2.4 DIV

除法运算（÷）

公式：变量类型（变量名）=变量类型（变量名）÷变量值来源（参数）

若要对全局整数或全局数值变量进行计算，则在变量类型处选择 GINT 或 GDOUBLE。

其中变量值来源若选择自定义，则参数则可以手动填写在“新参数”处。同时也可以为其它变量值。

例
前提：GD001=3.4 GI001=2
指令：DIV GD001 GI001
意义：GD001=GD001÷GI001
结果：GD001=1.7

5.3.4.2.5 MOD

取余运算（MOD）

公式：变量类型（变量名）=变量类型（变量名）MOD 变量值来源（参数）

若要对全局整数或全局数值变量进行计算，则在变量类型处选择 GINT 或 GDOUBLE。

其中变量值来源若选择自定义，则参数则可以手动填写在“新参数”处。同时也可以为其它变量值。

例
前提：GD001=14 GI001=3
指令：MOD GD001 GI001
意义：GD001=GD001 MOD GI001
结果：GD001=2（14÷3=4 余 2）

5.3.4.3 直接变量赋值

通过 SETBOOL、SETINT、SETDOUBLE 指令可以在运行程序时直接改变变量的值。

1. 在程序中，点击“插入”按钮；
2. 选择“变量类”；
3. 若要改变全局 BOOL 型变量，则选择 SETBOOL 指令，点击确定；
4. 变量类型处选择“GBOOL”；变量名选择之前定义过的全局 BOOL 变量；变量值来源选择“自定义”；新参数处填写需要改变为的值，若需要将该变量值改为 1，则在此处填入 1；

例如，需要在运行程序时将 GB001 变量的值改为 1，则按照下图所示填写参数。

SETINT、SETDOUBLE 分别用来设置 INT、DOUBLE 类型变量，用法同上。

5.3.4.4 使用全局数值变量来计数

在程序运行过程中，所有的计算、赋值操作均是对缓存中的数值进行更改的，并不会对“变量-全局数值”界面中的值进行修改，即当程序运行停止后所有全局变量的值都会还原。

若要对某一循环过程（如 WHILE 内循环）进行计数，则可以使用 FORCESET 指令。

使用场景：某一 WHILE 和 ENDWHILE 指令之间为一个工序，在该内部有一条 ADD GI001 1 指令，即每一次在 WHILE 和 ENDWHILE 之间循环，GI001 变量的值均加一，即该工序执行次数加一，在程序运行停止后，GI001 的数值还原为 0，无法查看该工序运行次数。

解决方案：在 Add GI001 1 指令之后插入一个 FORCESET GI001 指令。当程序运行结束后，进入“变量-全局数值”界面可看到 GI001 的值即代表该工序的运行次数。

插入方法：

1. 在“程序”界面点击【插入】按钮；
2. 选择“变量类”-“FORCESET”，点击“确定”；
3. 选择变量类型，若要改变全局整数型变量，则选择 GINT，变量名选择“GI001”；
4. 点击【插入】按钮，完成。

5.4 局部数值变量

局部数值变量仅能用于所定义的程序本身，如程序 A 的变量在程序 B 中不能使用。

数值型变量储存的是数值，包含了整数型变量、实数型变量、布尔型变量三种。定义的所有局部数值变量都只能用于当前程序，其他程序、后台程序都无法使用。

5.4.1 局部变量使用

5.4.1.1 定义局部数值变量

定义局部变量与定义全局变量的方法不同。定义局部变量需要在***程序页面***点击变量-局部变量页面设置。

整型 I

局部整数变量，用来存储整数型变量。变量名为 Ixxx。

默认为 0，选中需要修改的变量名点击修改，输入数值后点击保存。

浮点型 D

局部实数变量，用来存储实数型变量。变量名为 Dxxx。

默认为 0，选中需要修改的变量名点击修改，输入数值后点击保存。

布尔型 B

局部布尔变量，用来存储布尔型变量。变量名为 Bxxx。

默认为 0，选中需要修改的变量名点击修改，输入数值后点击保存。

5.4.1.2 使用计算指令为局部变量赋值

使用 ADD、SUB、MUL、DIV、MOD 指令对局部变量进行计算并赋值的方法和对全局变量的计算方法相同。

5.4.1.3 直接为变量赋值

使用 SETINT、SETDOUBLE、SETBOOL 指令对局部变量直接赋值的方法和对全局变量进行直接赋值的方法相同。

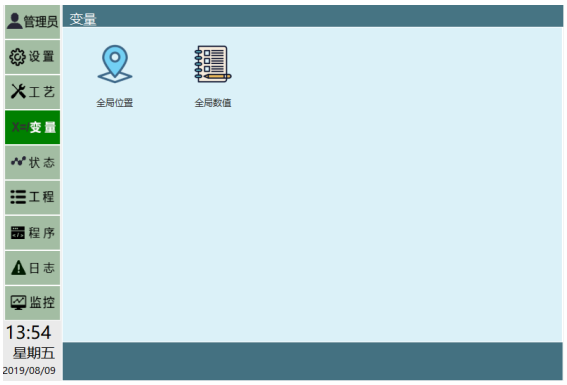
第6章 位置变量

本章主要说明本控制系统的变量设置的相关情况。

全局位置变量	全局位置、G	999 个	G001
局部位置变量	局部位置 P 点	每个作业文件 999 个	P001
	局部位置 E 点		E001
	局部位置 S 点(IMOV)		S001
	局部位置 R 点(SAMOV)		R001

6.1 全局位置变量

全局位置变量（G）在一个机器人的所有作业文件中均可使用。定义全局位置变量需要在“变量-全局位置”界面进行。



全局位置变量定义方法如下：

1. 进入“变量”-“全局位置”界面；
2. 选中需要定义的变量，如 G001；
3. 示教机器人到需要定义的位置，并切换坐标系到需要的坐标系，如直角坐标系；
4. 点击【修改】按钮；
5. 点击【记录当前点】按钮；

6. 点击【保存】按钮。

6.2 局部位置变量

局部位置变量（P）仅能用于单独的一个作业文件，不能在所有的作业文件之间通用。

局部位置变量的定义仅在插入 MOVJ、MOVL、MOVC 等运动类指令时，选择“新建”变量时自动定义。

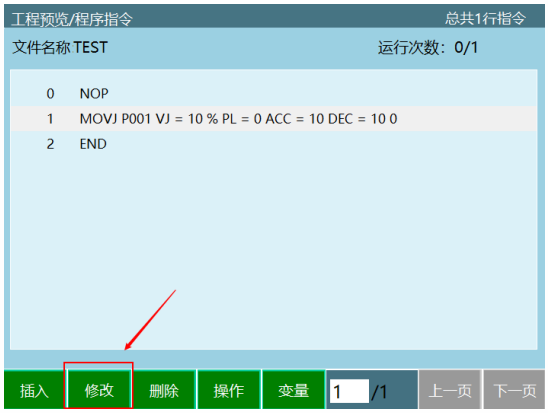
局部位置变量设置方法 1

1.点击程序-变量-局部变量进入局部变量查看界面

2.可以对局部位置变量进行，修改点位、增加点位、运行到该点、写入当前位置等功能

局部位置变量设置方法 2

1. 新建或修改 MOVJ 指令，进入指令界面



2. 当前位置列显示当前选中的坐标系下的机器人位置；P001 列显示 P 点选中坐标系下的机器人位置

3. 将机器人移动到 P 点，需示教模式上电操作；

将当前位置设置为 P 点，点击后把当前点位保存到局部 P 点；

手动修改，打开可手填 P 点坐标。

6.3 位置变量计算类指令的使用

6.3.1 POSADD 指令

位置变量加法运算（+），该指令能够对位置变量（全局、局部）单一轴的值进行加法运算，然后再赋值给该轴。

位置变量的变量名可以为数值变量，如 I001=50，那么 P$I001 则为 PI001.

该指令能够对位置变量的单一轴在任意坐标系下进行加法运算，不论该位置变量在插入时为何种坐标系，但在赋值时会转换为原有坐标系。例如，对 P001 变量的第二个轴进行加法运算，P001 坐标为关节坐标系下的（0,0,0,0,0,0），需要对该点在 Z 轴加 10，则在计算时会将 P001 转换为直角坐标（500,0,1000,0,0,0），再对 Z 轴加 10，即（500,0,1010,0,0,0），最后再转换为关节坐标（0,-1,1,0,1,0），并将该值赋给 P001。

公式：位置变量=位置变量[坐标系(轴）]+数值变量或数字

若要对全局整数或全局数值变量进行计算，则在变量类型处选择 GINT 或 GDOUBLE。

其中变量值来源若选择手填，则参数则可以手动填写在“新参数”处。同时也可以为其它变量值。

6.3.2 POSSUB 指令

位置变量减法运算（-），该指令能够对位置变量（全局、局部）单一轴的值进行减法运算，然后再赋值给该轴。

该指令意义与方法类似于 POSADD 指令。

公式：位置变量=位置变量[坐标系(轴）]-数值变量或数字

若要对全局整数或全局数值变量进行计算，则在变量类型处选择 GINT 或 GDOUBLE。

其中变量值来源若选择手填，则参数则可以手动填写在“新参数”处。同时也可以为其它变量值。

6.3.3 POSSET 指令

位置变量赋值，该指令能够对位置变量（全局、局部）单一轴的值直接进行赋值。

该指令意义与方法类似于 POSADD 指令。

公式：位置变量[坐标系(轴）]=数值变量或数字

若要对全局整数或全局数值变量进行计算，则在变量类型处选择 GINT 或 GDOUBLE。

其中变量值来源若选择手填，则参数则可以手动填写在“新参数”处。同时也可以为其它变量值。

6.3.4 READPOS 指令

读取位置变量坐标指令，该指令能够将位置变量的坐标值读取到数值变量中。

当选择读取“当前位置”坐标值时，所读取的为机器人运行时运行到该位置时的坐标值。

公式：数值变量（I、D、GI、GD）=位置变量[坐标系(轴)]

6.3.5 USERFRAME_SET 指令

修改用户坐标系指令， 该指令允许用户修改用户坐标系参数的某一轴的值。当修改后所有使用用户坐标系的点位均进行偏移。

如 P001、P002、P003 均使用用户坐标系 1，插入 USERFRAME_SET 指令，令用户坐标系 1 的 X 参数加 10，则 P001、P002、P003 三个位置变量均向 X 轴偏移 10mm。

6.3.6 TOOLFRAME_SET 指令

修改工具坐标指令，该指令可以修改工具坐标系某一轴的值。修改后，使用程序中的轨迹，会随着工具坐标系值的修改而变化。

例如，原工具偏移为（0,0,200,00,0），使用该指令修改 Z 轴方向偏移为 100，则运行中 6 轴法兰中心位置会往下偏移 100mm，若是换为对应的工具手，Z 轴偏移 200mm 的工具手换为 Z 轴偏移 100mm 的工具手，尖端位置不变。

6.3.7 COPYPOS 指令

复制点位指令，把当前位置、局部位置变量、全局位置变量等复制到另一局部或全局位置变量中。

例如把当前位置复制到局部位置变量，源位置变量类型：当前位置，源位置变量名：不选，目标位置变量类型：局部位置变量，目标位置变量名：P001 即可

6.4 **形态参数**

形态参数仅 6 轴机器人可用。

形态值为机器人 1 轴、3 轴、5 轴位置的 10 进制转换值

转换方式

例如某个六轴机器人 1 轴为 59 度、2 轴为 69 度、3 轴为 79 度、4 轴为 89 度、5 轴为 99 度、6 轴为 109 度;

取其中的 1/3/5 轴，点位范围在-90~+90 之间为 1，不在为 0；

所以结果如下

轴	1 轴	3 轴	5 轴
二进制数值	0	0	1

二进制数 001=十进制 1

形态值为十进制结果再加 1，该点位形态值为 2。

6.5 工具手参数

设置直角坐标、工具坐标、用户坐标点位绑定的工具手，不绑定选择无；若运动时工具手和点位参数不同，则无法运行。

例如，绑定工具手 2，使用工具手单步运行使用该点的指令，会报错。

6.6 用户坐标参数

设置用户坐标点位绑定用户坐标，不绑定选择无；若运动时用户坐标和点位参数不同，则无法运行。

例如，绑定用户坐标 1，使用用户坐标 5 单步运行使用该点的指令，会报错。

6.7 程序局部点参数说明

该功能介绍程序中点位保存的格式。

```
//DIR
//JOB
//NAME XXX
//POS
///NPOS 2,0,0,0,0,0
///POSTYPE PULSE
///PULSE
P001 = 0,0,0,0,0,0,0,11,22,33,44,55,66,0
P002 = 1,1,0,0,0,0,0,815,0,1297,3.1416,0,0,0
```

例如 P002 = 1,1,0,0,0,0,0,815,0,1297,3.1416,0,0,0

点位数据分解如下:

P002	点位名	点位名 P001-P999
1	坐标系	0：关节　　1：直角　　2：工具　　3：用户
1	角度/弧度	0：角度（关节点）1：弧度（直角点、工具点、用户点）
0	形态/左右手	六轴时为形态参数，四轴 SCARA 时为左右手参数
0	工具	工具手编号
0	用户	用户坐标编号
0	预留	预留
0	预留	预留
815	1 轴	点位 1 轴坐标
0	2 轴	点位 2 轴坐标
1297	3 轴	点位 3 轴坐标
3.1416	4 轴	点位 4 轴坐标
0	5 轴	点位 5 轴坐标
0	6 轴	点位 6 轴坐标
0	7 轴	点位 7 轴坐标

第7章 条件判断类指令的使用

条件判断类指令包含了 CALL、IF、WHILE、WAIT、JUMP 等指令。

7.1 指令说明

7.1.1 CALL

CALL 指令用来调用子程序。

本系统中在建立程序时没有区分主程序与子程序，当一个程序调用另一个程序时，被调用的程序则为子程序。

两个程序不能相互调用，即程序 A 调用程序 B 后，程序 B 不可调用程序 A。

参数	含义
程序名称	被调用的程序的程序名

例
前提：已建立 Job1、Job2 两个程序，在 Job1 中插入 CALL 指令
指令：CALL [Job2]
含义：调用子程序 Job2
过程：当 Job1 的指令运行到 CALL 指令时，程序跳转到程序 Job2，运行完程序 Job2 的所有指令后跳转回程序 Job1 中 CALL [Job2]指令的下一行指令继续运行。

7.1.2 IF

如果 IF 指令的条件满足时，则执行 IF 与 ENDIF 之间的指令，如果 IF 指令的条件不满足，则直接跳转到 ENDIF 指令继续运行 ENDIF 下面的指令，不运行 IF 与 ENDIF 之间的指令。

IF 的判断条件为（比较数 1 比较方式 比较数 2），例如比较数 1 为 2，比较数 2 为 1，比较方式为">"，则 2>1，判断条件成立；若比较方式为"<"或"=="，则判断条件不成立。

IF 指令可以单独使用，也可搭配 ELSEIF、ELSE 两条指令使用。***注意，ELSEIF、ELSE 指令不可脱离 IF 指令单独使用!***

注意，当程序的开头为 IF 且最后一行为 ENDIF 指令时，请在 IF 指令上方或 ENDIF 下方插入一条 0.1 秒的 TIMER（延时）指令，否则当 IF 指令的条件不满足时会导致程序陷入死机状态。

插入 IF 指令时会同时插入 ENDIF 指令，当删除 IF 指令时请注意将对应的 ENDIF 指令也删掉，否则会导致程序无法执行。

IF 指令中可以嵌套另一个 IF 指令或 WHILE、JUMP 等其它条件判断类指令。

参数	含义
参数类型	比较数 1 的类型，变量或数字、模拟量的输入值
参数名	若上一项选择的类型为变量（INT、DOUBLE、BOOL、GINT、GDOUBLE、GBOOL），则此处为比较数 1 的变量名 若上一项选择的类型为输入值（DIN、AIN），则此处为数字输入或模拟输入的端口号
比较方式	== 等于 < 小于 > 大于 <= 小于或等于 >= 大于或等于 != 不等于
变量值来源	比较数 2 的类型，自定义或变量或数字、模拟量的输入值
新参数	若上一项选择的类型为自定义，则此处不可选 若上一项选择的类型为变量（INT、DOUBLE、BOOL、GINT、GDOUBLE、GBOOL），则此处为比较数 1 的变量名 若上一项选择的类型为输入值（DIN、AIN），则此处为数字输入或模拟输入的端口号
来源参数	若变量值来源处选择的为自定义，则在此处直接填写比较数 2 的值

例 1
前提：已定义全局变量或局部变量，如 GI001=8
指令：IF(GI001<9)
　　其它指令，如 MOVJ 等
　　ENDIF
含义：如果 GI001<9，则运行 IF 与 ENDIF 之间的指令，若不满足则不运行
过程：因为 GI001=8<9，则条件成立，运行 IF 与 ENDIF 之间的指令，运行完后继续运行 ENDIF 下面的指令。
例 2
前提：已定义全局变量或局部变量，如 GI001=5，D001=8.88
指令：IF(GI001>=D001)
　　其它指令，如 MOVJ 等
　　ENDIF
含义：如果 GI001>=D001，则运行 IF 与 ENDIF 之间的指令，若不满足则不运行
过程：因为 GI001=5，D001=8.88，5<8.88，则条件不成立，不会运行 IF 与 ENDIF 之间的指令，程序跳转到 ENDIF 下面一行指令继续运行。
例 3
前提：已连接好外部的 IO 设备，如数字 IO 的端口 10 的输入值为 1
指令：IF(DIN10=1)
　　其它指令，如 MOVJ 等
　　ENDIF
含义：如果数字 IO 端口 10 的输入值=1，则运行 IF 与 ENDIF 之间的指令，若不满足则不运行
过程：因为数字 IO 的端口 10 的输入值为 1，即 DIN10=1，所以条件满足，运行 IF 与 ENDIF 之间的指令后继续运行 ENDIF 下面的指令。

7.1.3 ELSE

ELSE 指令必须插入在 IF 和 ENDIF 之间，但是一个 IF 指令只能嵌入一条 ELSE 指令。

当 IF 的判断条件成立时，会运行 IF 与 ELSE 之间的指令后跳转到 ENDIF 的下一行指令继续运行，而不运行 ELSE 和 ENDIF 之间的指令。

当 IF 的判断条件不成立时，会跳转到 ELSE 与 ENDIF 之间的指令运行，而不运行 IF 与 ELSE 之间的指令。

注意，当删除 IF 指令时，需删除与其对应的 ELSE 和 ENDIF 指令，否则会导致程序无法运行。

例 1
前提：已定义全局变量或局部变量，如 GI001=8
指令：IF(GI001<9)
　　其它指令 1，如 MOVJ 等
　　ELSE
　　其它指令 2，如 MOVJ 等
　　ENDIF
含义：如果 GI001<9，则运行 IF 与 ELSE 之间的指令 1，若不满足则运行 ELSE 和 ENDIF 之间的指令 2。
过程：因为 GI001=8<9，则条件成立，运行 IF 与 ELSE 之间的指令，运行完后继续运行 ENDIF 下面的指令。
例 2
前提：已定义全局变量或局部变量，如 GI001=5，D001=8.88
指令：IF(GI001>=D001)
　　其它指令 1，如 MOVJ 等
　　ELSE
　　其它指令 2，如 MOVJ 等
　　ENDIF
含义：如果 GI001>=D001，则运行 IF 与 ELSE 之间的指令 1，若不满足则运行 ELSE 和 ENDIF 之间的指令 2。
过程：因为 GI001=5，D001=8.88，5<8.88，则条件不成立，会运行 ELSE 与 ENDIF 之间的指令 2，之后继续运行 ENDIF 下面的指令。

7.1.4 ELSEIF

ELSEIF 指令必须插入在 IF 和 ENDIF 之间。ELSEIF 与 ENDIF 之间还可以插入一条 ELSE 指令或多条 ELSEIF 指令。

当 IF 的条件满足时，会忽略掉 ELSEIF 和 ELSEIF 与 ENDIF 之间的指令，仅运行 IF 与 ELSEIF 之间的指令，然后跳转到 ENDIF 下面的一行指令继续运行。

当 IF 的条件不满足时，会跳转到 ELSEIF 指令，判断 ELSEIF 的判断条件，若满足，则运行 ELSEIF 和 ENDIF 之间的指令，然后继续运行 ENDIF 下面的指令；若不满足，则直接跳转到 ENDIF 下面的一行指令继续运行。

若在 IF 与 ENDIF 中嵌套了多条 ELSEIF，当 IF 的判断条件不成立时首先判断第一条 ELSEIF 的判断条件，若成立则运行第一条 ELSEIF 与第二条 ELSEIF 之间的指令；若不成立则判断第二条 ELSEIF 的判断条件，以此类推。

注意，当删除 IF 指令时，需删除与其对应的 ELSEIF 和 ENDIF 指令，否则会导致程序无法运行。

例 1
前提：已定义全局变量或局部变量，如 GI001=8
指令：IF(GI001<9)
　　其它指令 1，如 MOVJ 等
　　ELSEIF(GI001>7)
　　其它指令 2，如 MOVJ 等
　　ENDIF
含义：如果 GI001<9，则运行 IF 与 ELSEIF 之间的指令 1，若不满足则判断 ELSEIF 的判断条件，若满足则运行其它指令 2，若不满足则跳转到 ENDIF 下面的指令继续运行。
过程：因为 GI001=8<9，则条件成立，运行 IF 与 ELSEIF 之间的指令，运行完后继续运行 ENDIF 下面的指令。
例 2
前提：已定义全局变量或局部变量，如 GI001=5，D001=8.88
指令：IF(GI001>=D001)
　　其它指令 1，如 MOVJ 等
　　ELSEIF(D001<9)
　　其它指令 2，如 MOVJ 等
　　ENDIF
含义：如果 GI001>=D001，则运行 IF 与 ELSE 之间的指令 1，若不满足则判断 ELSEIF 的判断条件，若满足则运行其它指令 2，若不满足则跳转到 ENDIF 下面的指令继续运行。
过程：因为 GI001=5，D001=8.88，5<8.88，则条件不成立，判断 ELSEIF 的判断条件，因为 D001=8.88<9，条件成立，则运行其它指令 2。
例 3
前提：已定义全局变量或局部变量，如 GI001=5，D001=8.88
指令：IF(GI001>=D001)
　　其它指令 1，如 MOVJ 等
　　ELSEIF(D001>9)
　　其它指令 2，如 MOVJ 等
　　ELSE
　　其它指令 3，如 MOVJ 等
　　ENDIF
含义：如果 GI001>=D001，则运行 IF 与 ELSE 之间的指令 1，若不满足则判断 ELSEIF 的判断条件，若满足则运行其它指令 2，若不满足则运行 ELSE 和 ENDIF 之间的其它指令 3，然后继续运行 ENDIF 下面的指令。
过程：因为 GI001=5，D001=8.88，5<8.88，则条件不成立，判断 ELSEIF 的判断条件，因为 D001=8.88<9，条件不成立，则运行其它指令 3。

例 4
前提：已定义全局变量或局部变量，如 GI001=5，D001=8.88
指令：IF(GI001>=D001)
　　其它指令 1，如 MOVJ 等
　　ELSEIF(D001>9)
　　其它指令 2，如 MOVJ 等
　　ELSEIF(GI001<6)
　　其它指令 3，如 MOVJ 等
　　ELSEIF(GI001>4)
　　其它指令 4，如 MOVJ 等
　ENDIF
含义：如果 GI001>=D001，则运行 IF 与 ELSE 之间的指令 1，若不满足则判断第一条 ELSEIF 的判断条件，若满足 D001>9 则运行其它指令 2，若不满足则判断第二条 ELSEIF 的判断条件，若 GI001<6 则运行其它指令 3，若不满足则判断第三条 ELSEIF，以此类推。
过程：因为 GI001=5，D001=8.88，5<8.88，则条件不成立，判断 ELSEIF 的判断条件，因为 D001=8.88<9，条件不成立，判断第 2 条 ELSEIF，GI001=5<6，条件成立，则运行其它指令 3，然后跳转到 ENDIF 下面的指令继续运行。

7.1.5 WHILE

当 WHILE 指令的条件满足时，会循环运行 WHILE 与 ENDWHILE 两条指令之间的指令。在运行到 WHILE 指令之前若判断条件不满足，在运行到 WHILE 指令时会直接跳转到 ENDWHILE 指令而不运行 WHILE 与 ENDWHILE 之间的指令；若在运行 WHILE 与 ENDWHILE 之间的指令过程中，判断条件变成不满足，会继续运行，直到运行到 ENDWHILE 行，不再循环而是继续运行 ENDWHILE 下面的指令。

WHILE 的判断条件为（比较数 1 比较方式 比较数 2），例如比较数 1 为 2，比较数 2 为 1，比较方式为">"，则 2>1，判断条件成立；若比较方式为"<"或"==，则判断条件不成立。

注意，插入 WHILE 指令的同时会同时插入 ENDWHILE 指令。若要删除 WHILE 指令请同时删掉其对应的 ENDWHILE 指令，否则会导致程序无法运行。

当程序的开头为 WHILE 且最后一样指令为 ENDWHILE 时，请在程序的开头或结尾插入一条 0.3 秒的 TIMER（延时）指令。否则当 WHILE 指令的条件不满足时会导致程序陷入死机。

当 WHILE 内部的指令没有运动类指令或在某种情况下可能会陷入死循环时，请在 WHILE 与 ENDWHILE 间插入一条 0.3 秒的 TIMER（延时）指令，否则当 WHILE 指令的条件满足时可能会导致程序陷入死机。

WHILE 指令可以同时嵌套多个 WHILE、IF 或 JUMP 等其它判断类指令使用。

参数	含义
参数类型	比较数 1 的类型，变量或数字、模拟量的输入值
参数名	若上一项选择的类型为变量（INT、DOUBLE、BOOL、GINT、GDOUBLE、GBOOL），则此处为比较数 1 的变量名 若上一项选择的类型为输入值（DIN、AIN），则此处为数字输入或模拟输入的端口号
比较方式	== 等于 < 小于 > 大于 <= 小于或等于 >= 大于或等于 != 不等于
变量值来源	比较数 2 的类型，自定义或变量或数字、模拟量的输入值
新参数	若上一项选择的类型为自定义，则此处不可选 若上一项选择的类型为变量（INT、DOUBLE、BOOL、GINT、GDOUBLE、GBOOL），则此处为比较数 1 的变量名 若上一项选择的类型为输入值（DIN、AIN），则此处为数字输入或模拟输入的端口号
来源参数	若变量值来源处选择的为自定义，则在此处直接填写比较数 2 的值

例 1
前提：已经定义了变量 GI001=1
指令：WHILE(GI001<2)
　　其它指令
　　ENDWHILE
含义：当 GI001<2 时，会循环运行 WHILE 与 ENDWHILE 之间的其它指令。直到条件不成立时，运行到 ENDWHILE 指令不会再循环而是继续运行 ENDWHILE 下面的指令。
过程：因为 GI001=1<2，会循环运行 WHILE 与 ENDWHILE 之间的其它指令。直到条件不成立时，运行到 ENDWHILE 指令不会再循环而是继续运行 ENDWHILE 下面的指令。
例 2
前提：已经定义了变量 GI001=1,D001=7
指令：WHILE(GI001<2)
　　其它指令 1，MOVJ 等
　　WHILE(D001<10)
　　其它指令 2，MOVJ 等
　　ADD D001 1
　　ENDWHILE
　　其它指令 3
　　ADD GI001 1
　　ENDWHILE
含义：当 GI001<2 时，会循环运行 WHILE 与 ENDWHILE 之间的所有指令，在运行到 WHILE（D001<10）时，判断 D001<10，若成立则循环运行其它指令 2 和 ADD 指令，直到 D001>=10 时，跳出中间的 WHILE 指令，继续运行其它指令 3 和 ADD 指令，再循环，直到 GI001>=2 时跳出 WHILE。
过程：开始 GI001=1<2，D001=7<10，所以一开始两个 WHILE 指令的判断条件均成立，会循环运行 WHILE(D001<10)和中间 ENDWHILE 之间的其它指令 2 和 ADD 指令，每循环一次 D001 会加 1，循环 3 次后，D001=10，中间的判断条件不成立，继续运行其它指令 3 和 ADD GI001 1 指令，每循环一次 GI001 加 1，运行 1 次后 GI001=2，条件不成立，继续运行 ENDWHILE 下面的指令。

7.1.6 WAIT

WAIT 即等待，可以选择是否有等待时间。当没有勾选“TIME”选项，则在判断条件不成立时一直停留在该 WAIT 指令等待，直到判断条件成立。若勾选了“TIME”选项，则会在等待该参数的时长后不再等，继续运行下一条指令。若在等待时条件变为成立，则立刻运行下一条指令。

参数	含义
参数类型	比较数 1 的类型，变量或数字、模拟量的输入值

参数名	若上一项选择的类型为变量（INT、DOUBLE、BOOL、GINT、GDOUBLE、GBOOL），则此处为比较数 1 的变量名 若上一项选择的类型为输入值（DIN、AIN），则此处为数字输入或模拟输入的端口号
比较方式	== 等于 < 小于 > 大于 <= 小于或等于 >= 大于或等于 != 不等于
变量值来源	比较数 2 的类型，自定义或变量或数字、模拟量的输入值
新参数	若上一项选择的类型为自定义，则此处不可选 若上一项选择的类型为变量（INT、DOUBLE、BOOL、GINT、GDOUBLE、GBOOL），则此处为比较数 1 的变量名 若上一项选择的类型为输入值（DIN、AIN），则此处为数字输入或模拟输入的端口号
来源参数	若变量值来源处选择的为自定义，则在此处直接填写比较数 2 的值
TIME	可选项，不选则永远等待直到条件成立 选择则可填写等待时间（秒），等待到该时长后，即使条件依然不成立，依然会跳转到下一行继续运行。
是否连续	如果选择“是”，则在运行该指令前条件满足时，上一条指令的 PL 与该指令下一条的 PL 可以连续。如果选择否则会打断 PL。

例
前提：已经定义了变量 GI001=1
指令：WAIT（GI001==2）T = 2 NOW = 1
含义：当 GI001 不等于 2 时，程序停留在这一条指令等待，但是等待超过两秒后将不再等，跳到下一条程序继续运行。在等待过程中若条件满足则立即跳转到下一行继续运行。
过程：因为 GI001 不等于 2，程序停留在这一条指令等待，但是等待超过两秒后将不再等，跳到下一条程序继续运行。

7.1.7 LABEL

LABEL 标签指令，需要和 JUMP 指令配合指令配合使用，单独的 LABEL 指令无意义。

参数	含义
标签名	字符开头的字符串，长度最大 8 个字符

7.1.8 JUMP

JUMP 用于跳转，***必须与 LABEL（标签）指令配合使用***。

JUMP 可以设置有无判断条件。***当设置为没有判断条件时，运行到该指令会直接跳转到对应的 LABEL 指令后继续运行 LABEL 下一行指令***。

当设置为有判断条件时，***若条件满足则跳转到 LABEL 指令行；若条件不满足则忽略 JUMP 指令，继续运行 JUMP 指令的下一行指令***。

LABEL 标签可以插在 JUMP 的上方或者下方，但***不可跨程序跳转***。

LABEL 标签名必须为字母开头的两位以上字符。

插入 LABEL 标签对程序的运行没有影响，但是要符合程序运行规则，例如不能插在 MOVC 指令的上面或插在局部变量定义指令的上面。

参数	含义
标签名	已插入 LABEL 指令的标签名，选项
判断条件	选项，若选中则可以设置判断条件 若不选中则运行到 JUMP 后直接跳转
参数名	若上一项选择的类型为变量（INT、DOUBLE、BOOL、GINT、GDOUBLE、GBOOL），则此处为比较数 1 的变量名 若上一项选择的类型为输入值（DIN、AIN），则此处为数字输入或模拟输入的端口号

比较方式	== 等于 < 小于 > 大于 <= 小于或等于 >= 大于或等于 != 不等于
变量值来源	比较数 2 的类型，自定义或变量或数字、模拟量的输入值
新参数	若上一项选择的类型为自定义，则此处不可选 若上一项选择的类型为变量（INT、DOUBLE、BOOL、GINT、GDOUBLE、GBOOL），则此处为比较数 1 的变量名 若上一项选择的类型为输入值（DIN、AIN），则此处为数字输入或模拟输入的端口号
来源参数	若变量值来源处选择的为自定义，则在此处直接填写比较数 2 的值

例 1
前提：无
指令：MOVJ
　　LABEL *C1
　　其它指令 1，MOVJ 等
　　JUMP *C1
　　其它指令 2
含义：运行到 JUMP 指令后跳转到 LABEL *C1 行继续运行其它指令 1。
过程，运行到 JUMP 指令后跳转到 LABEL *C1 行继续运行其它指令 1。
例 2
前提：已定义变量 I001=1
指令：MOVJ
　　LABEL *C1
　　其它指令 1，MOVJ 等
　　JUMP *C1 WHEN（I001==0）
　　其它指令 2
含义：运行到 JUMP 指令时进行判断，若 I001 等于 0，则跳转到 LABEL *C1 行运行其它指令 1，若条件不成立则不跳转，继续运行其它指令 2。
过程：因为 I001=1 不等于 0，所以不会跳转。

7.1.9 UNTIL

UNTIL 指令用于在一个运动过程中跳出。即在机器人的一个运动过程中暂停并开始下一个过程。当条件满足时，不论当前机器人是否运行，立即暂停并开始 ENDUNTIL 指令下面的一条指令。

UNTIL 的判断条件为（比较数 1 比较方式 比较数 2），例如比较数 1 为 2，比较数 2 为 1，比较方式为">"，则 2>1，判断条件成立；若比较方式为"<"或"=="，则判断条件不成立。

注意，***插入 UNTIL 指令的同时会同时插入 ENDUNTIL 指令。若要删除 UNTIL 指令请同时删掉其对应的 ENDUNTIL 指令，否则会导致程序无法运行***。

参数	含义
参数类型	比较数 1 的类型，变量或数字、模拟量的输入值
参数名	若上一项选择的类型为变量（INT、DOUBLE、BOOL、GINT、GDOUBLE、GBOOL），则此处为比较数 1 的变量名 若上一项选择的类型为输入值（DIN、AIN），则此处为数字输入或模拟输入的端口号
比较方式	== 等于 < 小于 > 大于 <= 小于或等于 >= 大于或等于 != 不等于
变量值来源	比较数 2 的类型，自定义或变量或数字、模拟量的输入值
新参数	若上一项选择的类型为自定义，则此处不可选 若上一项选择的类型为变量（INT、DOUBLE、BOOL、GINT、GDOUBLE、GBOOL），则此处为比较数 1 的变量名 若上一项选择的类型为输入值（DIN、AIN），则此处为数字输入或模拟输入的端口号
来源参数	若变量值来源处选择的为自定义，则在此处直接填写比较数 2 的值

例
前提：已经定义了变量 GI001=1
指令：UNTIL(GI001<2)
　　其它指令
　ENDUNTIL
　MOVJ P003
含义：当运行 UNTIL 与 ENDUNTIL 之间的“其它指令”时，若 GI001 变成了<2 的数值，则暂停当前动作，跳转到 MOVJ P003 指令；若 GI001 始终>2，则运行完其它指令后再运行 MOVJ P003 指令。

7.1.10 CRAFTLINE

工艺跳行指令，配合专用工艺使用，配合专用工艺跳行使用

参数	含义
新参数	填写专用工艺程序行数

7.1.11 CMDNOTE

指令注释，可以使用该指令在程序适当位置添加注释，便于调试

参数	含义
注释内容	支持中英文

若注释内容为“INEXBOT 九众九”，则在程序界面该指令显示为##INEXBOT 九众九$$.

7.1.12 POS_REACHABLE

到达判断指令，用于判断目标点是否能到达，点位能够到达变量置 1，不能到达置 0

参数	含义
位置变量名	可选择 P 点、G 点
运动类型	可选择 MOVJ、MOVL

状态存入变量类型	可存入 BOOL、GBOOL
状态存入变量名	BOOL、GBOOL 变量名称

例
前提：已定义 BOOL 变量 A001、已定义位置变量 P001
指令：POS_REACHABLE MOVJ P001 A001
含义：计算能否使用 MOVJ 插补运行到 P001 位置，可以到达 A001 值为 1，不可以到达 A001 值为 0。

7.1.13 CLKSTART

CLKSTART 指令用于计时。运行该指令开始计时，并将时间记录到一个局部或者全局 DOUBLE 变量中。

参数	含义
序号	计时器的序号，可以同时使用 32 个计时器分别计时。
存入变量类型	将计时的时间存入到局部 DOUBLE 变量或者全局的 GDOUBLE 变量。
存入变量名	将时间存入的变量的变量名。

7.1.14 CLKSTOP

CLKSTOP 指令用于停止对应序号的计时器计时。停止后已存入变量的值不会归零。

参数	含义
序号	要停止计时的计时器的序号。

7.1.15 CLKRESET

CLKRESET 指令用于将对应序号的计时器归零。若没有使用该指令，下次运行 CLKSTART 指令会累积计时。

参数	含义
序号	要归零计时的计时器的序号。

第8章 多线程

8.1 局部后台任务编程

需要在后台任务中运行的程序需要在“设置-后台任务”中进行，其编程与编写普通程序相同。

注意

在 WHILE 循环中和整个程序的最后一行最好各插入一条 0.2s 的延时。

当编辑后台任务程序时，若要调试仅提供“STEP”单步运行的方式。若要整体运行调试，请在主程序中插入 PTHREAD_START 指令并运行来进行调试。

后台任务开启一次只执行一次，如需循环判断可以配合 WHILE 指令使用。

8.2 全局后台任务编程

需要在后台任务中运行的程序需要在“设置-后台任务”中进行，其编程与局部后台相同。

设置全局后台方式：

1. 全局后台程序编辑完成
2. 点击全局后台操作区域的操作-设为自启

3. 程序列表上方显示：“开机自启动：程序名”

4. 立即启动点击操作区内的启动按钮，否则重启后自动启动

可在监控-程序运行-后台任务中查看全局后台运行情况

8.3 主程序编程

在主程序中若要运行后台任务，需要在程序中插入 PTHREAD_START（开启线程）指令。需要退出后台任务请插入 PTHREAD_END（退出线程）指令。

后台任务仅在运行 PTHREAD_START 之后开始运行，主程序暂停时后台程序不暂停。

后台任务停止条件：

1. 程序运行到 PTHREAD_END 指令；

2. 程序停止，机器人停止使能。

3. 后台任务运行到结束行 END。

8.4 程序控制类指令

8.4.1 PTHREAD_START（开启线程）

运行 PTHREAD_START 指令则开启后台任务。该指令位于程序控制类指令中。

插入该指令时，点击【值】输入框，会自动弹出已建立的后台任务列表，选择需要运行的后台任务，点击【确定】按钮，则会选中该程序。

运行主程序时，当运行到 PTHREAD_START 指令，则开启后台任务。

8.4.2 PTHREAD_END（关闭线程）

运行 PTHREAD_END 指令会退出已运行的相应后台任务。

其插入与修改方法与 PTHREAD_START 相同。

8.4.3 PAUSERUN （暂停线程）

运行 PAUSERUN 指令会暂停全部任务、主程序、或者后台任务。

插入该指令时，点击类型的【值】下拉框，选择控制的类型，点击程序的【值】输入框，会自动弹出已建立的后台任务列表，选择需要运行的后台任务，点击【确定】按钮，则会选中该程序，全部、主程序时不可选。

注：按示教盒停止键仅暂停主程序。

运行程序时，当运行到 PAUSERUN 指令，则暂停设置的全部任务、主程序、或者后台任务。

8.4.4 CONTINUERUN（继续线程）

运行 CONTINUERUN 指令会继续运行主程序、或者后台任务。

插入该指令时，点击类型的【值】下拉框，选择控制的类型，点击程序的【值】输入框，会自动弹出已建立的后台任务列表，选择需要运行的后台任务，点击【确定】按钮，则会选中该程序，主程序时不可选。

运行程序时，当运行到 CONTINUERUN 指令，则继续运行主程序、或者后台任务。

8.4.5 STOPRUN （停止运行）

运行 STOPRUN 指令会停止运行所有任务。

该指令直接点击确认插入即可，不需要设置参数

8.4.6 RESTARTRUN（重新运行）

运行 RESTARTRUN 指令会重新运行所有任务。

该指令直接点击确认插入即可，不需要设置参数

8.5 后台任务支持的指令

后台任务的程序中当前仅支持以下指令:

类别	指令	内容
输入输出类	**DIN**	IO 输入
	DOUT	IO 输出
	AIN	模拟输入
	AOUT	模拟输出
	READ_DOUT	读取输出
定时器类	**TIMER**	延时
运算类	**ADD**	加
	SUB	减
	MUL	乘
	DIV	除
	MOD	模
	SIN	正弦

	COS	余弦
	ATAN	反正切
	LOGICAL_OP	逻辑运算
条件控制类	IF	如果
	ELSEIF	否则如果
	ELSE	否则
	WAIT	等待
	WHILE	内循环
	LABEL	标签
	JUMP	跳转
	CLKSTART	计时开始
	CLKSTOP	计时结束
	CLKRESET	计时复位
变量类	SETINT	赋值整型
	SETDOUBLE	赋值浮点型
	SETBOOL	赋值布尔型
	FORCESET	写入文件
通讯类	SENDMSG	发送数据
	PARSEMSG	解析数据
	READCOMM	读取
	OPENMSG	打开数据
	CLOSEMSG	关闭数据
	PRINTMSG	输出数据
	MSG_CONN_ST	获取信息连接状态
位置变量类	USERFRAME_SET	用户坐标修改
	TOOLFRAME_SET	工具坐标修改
	READPOS	读取点位
	POSADD	点位加
	POSSUB	点位减

	POSSET	点位改
	COPYPOS	复制点位
坐标切换类	SWITCHUSER	切换用户坐标
程序控制类	PAUSERUN	暂停运行
	CONTINUERUN	继续运行
	STOPRUN	停止运行
	RESTARTRUN	重新运行
	PTHREAD_START（局部后台不可用）	开启线程
	PTHREAD_END（局部后台不可用）	关闭线程
视觉命令类	VISION_RUN	开始视觉
	VISION_TGR	触发视觉
	VISION_POSNUM	获取视觉位置个数
	VISION_POS	获取视觉位置
	VISION_CLEAR	清除视觉位置信息
	VISION_END	结束视觉

注：运行模式按暂停按钮、远程模式 IO 暂停只暂停主程序，不暂停后台任务