

KUKA



Training
机器人编程 1
KUKA System Software 8.6
KUKA College



培训资料

发布日期: 16.06.2021
PROG P1 KSS 8 V3.2.2
KUKA Deutschland GmbH

©版权说明 2021

KUKA Deutschland GmbH
Zugspitzstraße 140
D-86165 Augsburg
德国

此文献或节选只有在征得 KUKA Deutschland GmbH 明确同意的情况下才允许复制或对外方开放。除了本文献中说明的功能外，控制系统还可能具有其他功能。但是在新供货或进行维修时，无权要求 KUKA Deutschland GmbH 提供这些功能。

我们已就印刷品的内容与描述的硬件和软件内容是否一致进行了校对。但是不排除有不一致的情况，我们对此不承担责任。但是我们定期校对印刷品的内容，并在之后的版本中作必要的更改。

我们保留在不影响功能的情况下进行技术更改的权利。

KIM-PS5-DOC

原版文件的译文

出版物: Pub COLLEGE PROG P1 KSS 8 (V3.2) (PDF-COL) zh
PB17011

书页构造: P1KSS8 - Roboterprogrammierung 1 (R3) V4.2
BS10350

版本: PROG P1 KSS 8 V3.2.2

目录

1	KUKA 机器人系统的结构和功能.....	9
1.1	学习单元：KUKA 机器人系统的结构和功能.....	9
1.2	机器人基础知识.....	9
1.3	KUKA 机器人的机械系统.....	10
1.4	机器人控制器 (V)KR C4.....	12
1.5	KUKA smartPAD.....	15
1.5.1	smartPAD 1 概览.....	16
1.5.2	smartPAD 2 概览.....	19
1.5.3	插上和拔下 smartPAD.....	21
1.5.4	smartPAD 基本设置	25
1.6	机器人编程.....	28
1.7	机器人安全.....	31
2	机器人系统文档.....	35
2.1	学习单元：机器人系统的文件.....	35
2.2	KUKA Xpert.....	35
2.3	KUKA Xpert URL.....	36
2.4	注册.....	37
2.5	登录.....	38
2.6	使用 KUKA Xpert Basic.....	39
2.7	练习：使用 KUKA Xpert Basic.....	41
3	机器人运动.....	43
3.1	学习单元：移动机器人.....	43
3.2	控制器信息读取查看.....	43
3.3	机器人控制系统的运行方式.....	46
3.3.1	停止反应.....	47
3.3.2	切换运行模式.....	49
3.4	机器人单轴移动.....	50
3.4.1	增量式手动运行.....	53
3.4.2	在紧急情况下不用控制系统移动机器人.....	54
3.4.3	练习：操作及轴相关的手动运行.....	57
3.5	与机器人相关的坐标系.....	58
3.5.1	坐标系和“右手法则”.....	59
3.6	在世界坐标系中点动机器人.....	61
3.6.1	练习：借助移动键在世界坐标系中操作以及手动运行.....	68
3.6.2	练习：借助 6D 鼠标在世界坐标系中操作以及手动运行.....	69
3.7	借助轨迹运行方式移动机器人.....	70
3.8	机器人位置的显示方法.....	73
3.8.1	在 smartPAD 上查询当前机器人位置.....	76
4	机器人的投入运行.....	79
4.1	学习单元：机器人投入运行.....	79
4.2	控制系统上的用户组.....	79
4.2.1	切换用户组.....	80
4.2.2	更改密码.....	82
4.2.3	在控制系统上管理权限.....	83

4.3	零点标定的原理.....	84
4.3.1	零点标定工具.....	87
4.3.2	确定机械零位.....	90
4.3.3	标定机器人零点.....	91
4.3.3.1	首次/标准零点标定.....	92
4.3.3.2	零点标定, 带负载校正.....	95
4.3.3.3	练习: 零点标定, 带偏移量的负载零点标定.....	101
4.3.4	小型机器人 AGILUS 上的零点标定.....	102
4.3.5	重要的零点标定文件.....	103
4.4	工具和基座管理.....	105
4.4.1	smartPAD 上的工具坐标和基坐标管理.....	106
4.5	机器人上的负载.....	109
4.5.1	工具负载数据.....	110
4.5.2	机器人上的附加负载.....	111
4.6	工具测量.....	114
4.6.1	添加新的工具/工件.....	118
4.6.1.1	TCP 测量 4 点法.....	120
4.6.1.2	XYZ 参照法.....	124
4.6.1.3	ABC-World 方法.....	127
4.6.1.4	ABC 2 点法.....	130
4.7	在工具坐标系中移动机器人.....	134
4.7.1	练习: 工具坐标系中的手动运行.....	138
4.8	测量基坐标.....	139
4.8.1	添加新的基坐标/固定工具.....	142
4.8.1.1	三点法.....	143
4.8.2	在基坐标系中点动机器人.....	148
4.8.2.1	练习: 基坐标系中的手动运行.....	152
4.9	借助对齐运行方式移动机器人.....	153
4.10	显示内存使用情况.....	156
4.11	控制系统关机.....	157
5	执行机器人程序.....	161
5.1	学习单元: 执行机器人程序.....	161
5.2	选择机器人程序.....	161
5.3	机器人程序看起来是怎样的?.....	163
5.4	程序初始化.....	164
5.5	进行程序启动.....	165
5.5.1	练习: 执行机器人程序.....	169
6	程序文件的使用.....	171
6.1	学习单元: 使用程序文件.....	171
6.2	创建程序模块.....	171
6.3	编辑程序模块.....	174
6.4	存档和恢复机器人程序.....	176
6.5	通过日志了解程序和状态变更.....	180
6.5.1	导出日志.....	182
7	建立及更改编程的运动.....	185
7.1	学习单元: 创建和更改编程的运动.....	185
7.2	创建新的运动指令需要哪些信息?.....	185

7.3	SPTP - 最优时间运动 (轴运动)	187
7.3.1	状态和转角方向 (Status & Turn)	189
7.3.2	SPTP 轨迹逼近	190
7.3.3	借助行指令对运动编程	191
7.3.3.1	行指令: SPTP	195
7.3.4	练习: 空运转程序 - 程序操作和 SPTP 运动	197
7.4	创建沿轨迹的运动	198
7.4.1	奇点	199
7.4.2	SLIN 编程	202
7.4.2.1	行指令: SLIN	204
7.4.2.2	SLIN 时的姿态引导	204
7.4.3	SCIRC 编程	206
7.4.3.1	行指令: SCIRC	210
7.4.3.2	轨迹运动时的轨迹逼近	211
7.4.3.3	SCIRC 时的姿态引导	211
7.4.3.4	SCIRC: 姿态引导举例: 辅助点	214
7.4.3.5	SCIRC: 姿态特性 - 目标点示例	216
7.4.3.6	执行 \$CIRC_MODE 时的限制	218
7.4.4	沿轨迹运行和轨迹逼近	219
7.5	用全局点编程	220
7.5.1	在概览中显示和更改全局点	222
7.6	更改运动命令	225
8	样条运动编程	233
8.1	学习单元: 样条运动编程	233
8.2	什么是样条?	233
8.3	样条组的说明	237
8.4	借助运动对样条组编程	239
8.4.1	样条组中的 SLIN	243
8.4.2	样条组中的 SCIRC	244
8.4.3	样条组中的 SPL	245
8.4.4	运动参数	246
8.5	编程提示	247
8.5.1	在样条组中更改点	247
8.5.2	样条曲线的速度说明	250
8.5.3	样条运动的语句选择	253
8.6	样条曲线的轨迹逼近	254
8.7	练习: 带样条组的轨迹轮廓	255
9	利用机器人程序中的逻辑功能	257
9.1	学习单元: 在机器人程序中使用逻辑函数	257
9.2	逻辑编程介绍	257
9.3	显示变量	258
9.3.1	显示输入端和输出端	259
9.3.2	显示旗标、计数器和计时器	261
9.4	简单的逻辑编程	262
9.4.1	等待功能的编程	262
9.4.2	单一信号输出功能编程	267
9.4.3	可选练习: 给轨迹运动配设逻辑	272

9.5	编程带样条和样条单个语句的逻辑.....	273
9.5.1	为样条触发器编程.....	273
9.5.1.1	样条曲线触发器编程.....	274
9.5.1.2	借助行指令给样条曲线触发器编程.....	277
9.5.1.3	编程提示.....	279
9.5.2	为条件停止编程.....	280
9.5.2.1	编程提示.....	281
9.5.2.2	条件停止编程.....	282
9.5.2.3	借助行指令编程条件停止.....	285
9.5.3	恒速运动区域.....	286
9.5.3.1	恒速运动区域编程.....	287
9.5.3.2	编程提示.....	291
9.5.4	练习：给样条运动配设逻辑.....	293
9.5.5	练习：恒速运动区域和条件停止.....	294
10	使用应用程序包.....	295
10.1	学习单元：使用工艺程序包.....	295
10.2	了解 GripperTech.....	295
10.3	用 KUKA.GripperTech 操作抓手.....	296
10.3.1	在 smartPAD 上显示抓手信号.....	297
10.4	配置抓手.....	299
10.4.1	给抓手进行参数设置.....	304
10.5	借助行指令编程抓手.....	306
10.5.1	借助行指令检查抓爪状态.....	309
10.5.2	练习：抓爪编程“标牌”.....	312
10.5.3	练习：抓爪编程“笔”.....	313
11	外部工具的配置和编程.....	315
11.1	学习单元：外部工具的配置和编程.....	315
11.2	机器人引导的工件和固定工具.....	315
11.2.1	测量固定工具.....	316
11.2.2	测量由机器人引导的工件.....	321
11.2.3	练习：测量外部工具和机器人引导的工件.....	326
11.3	固定工具点动.....	327
11.3.1	练习：用固定工具手动移动.....	332
11.4	用外部 TCP 进行运动编程.....	333
11.4.1	练习：用外部 TCP 进行运动编程.....	334
12	专家界面介绍.....	335
12.1	学习单元：专家界面介绍.....	335
12.2	借助模板创建程序.....	335
12.3	调整浏览器中的视图.....	337
12.4	调整编辑器中的视图.....	338
12.5	找到并排除程序中的错误.....	339
12.6	结构化机器人程序.....	341
12.6.1	给程序添加注释.....	341
12.6.2	缩进程序.....	344
12.6.3	隐藏程序行.....	344
12.6.4	链接机器人程序.....	345
12.6.5	练习：编程子程序调用.....	348

13	变量和协定.....	349
13.1	学习单元：变量和协议.....	349
13.2	KRL 中的数据管理.....	349
13.2.1	命名约定.....	349
13.2.2	数据类型.....	350
13.2.3	建立变量.....	350
13.2.4	变量的双重声明.....	351
13.2.5	变量的生存期和有效性.....	351
13.2.6	取决于存储位置的变量声明.....	351
13.2.7	KUKA 系统数据.....	352
13.3	简单数据类型变量.....	352
13.3.1	借助行指令对逻辑指令和循环编程.....	352
13.3.2	变量声明.....	353
13.3.3	简单数据类型变量的初始化.....	356
13.3.4	通过行指令声明并初始化变量.....	358
13.3.5	用 KRL 对简单数据类型的变量值进行操作.....	359
13.3.6	练习：简单的数据类型.....	364
14	使用程序流程控制.....	365
14.1	学习单元：程序运行控制的用处.....	365
14.2	给循环编程.....	365
14.2.1	Loop 循环编程.....	365
14.2.2	FOR 循环编程.....	368
14.2.3	WHILE 循环编程.....	370
14.2.4	REPEAT...UNTIL 循环编程.....	373
14.3	IF 语句编程.....	375
14.4	SWITCH- CASE 编程.....	378
14.5	编程跳转命令.....	380
14.6	用 KRL 语言给等待功能编程.....	381
14.6.1	时间等待功能.....	382
14.6.2	信号等待功能.....	383
14.7	练习：循环技术	387
15	使用上级控制系统作业.....	389
15.1	学习单元：使用上级控制系统作业.....	389
15.2	通过 PLC 启动程序的准备工作.....	389
15.3	修改链接 PLC 外部自动端口程序 (Cell.src).....	392
15.4	问题：上级控制系统上的工作	394
16	附录.....	395
16.1	自动运行外部接口的说明.....	395
16.1.1	自动运行外部接口的配置.....	401
16.1.2	练习：配置外部自动运行接口（手动）.....	405
16.2	缩略语表.....	406
	索引	409

1 KUKA 机器人系统的结构和功能

1.1 学习单元：KUKA 机器人系统的结构和功能

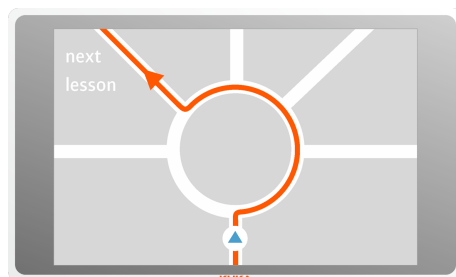


图 1-1：导航器

本学习单元讲授以下内容：

- 机器人技术基础知识
- KUKA 机器人的机械系统
- KR C4 机器人控制器
- KUKA smartPAD
- 机器人编程
- 机器人处的安全

1.2 机器人基础知识

什么是机器人？

机器人 *Roboter* 这个概念源于斯拉夫语中的字 *robota*，意为**重活**。工业机器人这个词的官方定义为：“机器人是一种可自由编程并受程序控制的设备。”控制系统、操作设备以及连接电缆和软件也同样属于机器人系统的范畴。

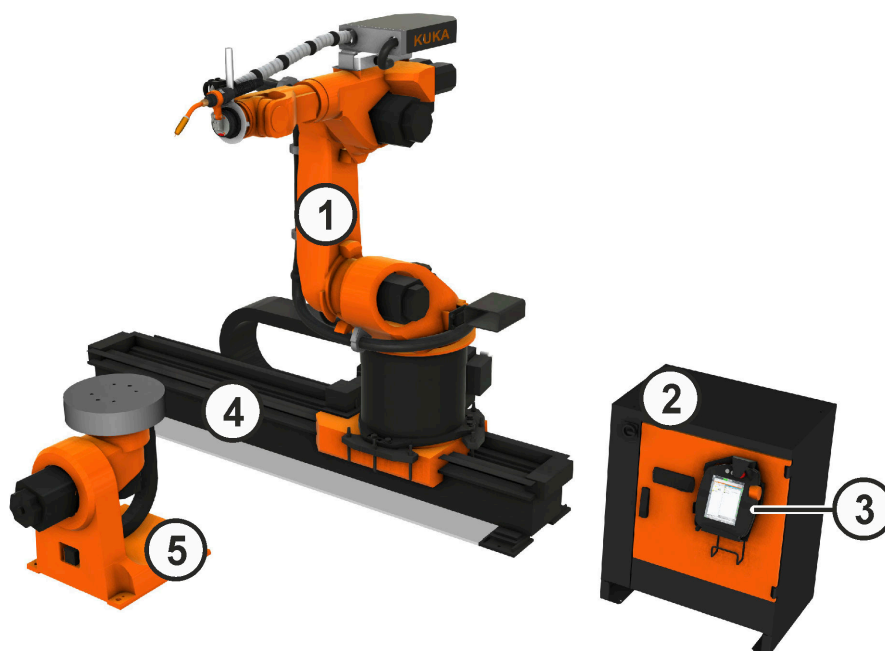


图 1-2：工业机器人

- 1 机械手（机器人机械系统）
- 2 控制系统（控制柜（V）KR C4）
- 3 示教器和编程器（KUKA smartPAD）
- 4 可选：KUKA 线性滑轨
- 5 可选：KUKA 双轴转台

所有不包括在工业机器人系统内的设备被称为**外围设备**，它们是：

- 工具（终端操作设备或工具）
- 防护装置
- 输送带，
- 传感器
- 其他机械
- 等等

1.3 KUKA 机器人的机械系统

什么是机械手？

机械手的机器人机械机构由一定数量的可移动的、相互链接的刚性关节组成，关节通过轴相互连接。可通过带前置齿轮箱的伺服电机精确移动轴。

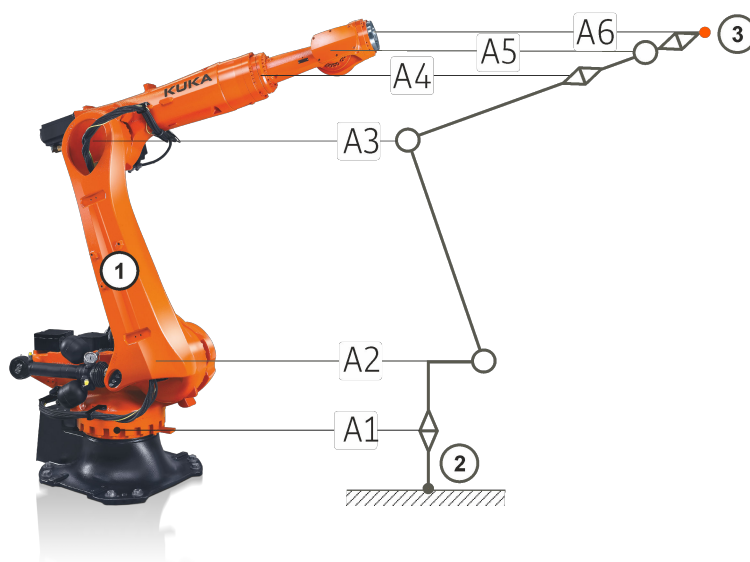


图 1-3：机械手

- 1 机械手（机器人机械系统）
 - 2 运动链的起点：机器人根部（ROBROOT）
 - 3 运动链的开放端：法兰（FLANGE）
- A1 机器人轴 1 至 6
- ..
- .
- A6

机器人的机械零部件概览

机器人机械系统的部件主要由铸铝和铸钢制成。在个别情况下也使用碳纤维部件。



图 1-4：机器人的机械零部件概览

- | | |
|---------------|------|
| 1 底座 | 5 大臂 |
| 2 转盘 | 6 小臂 |
| 3 软管套件（介质、能源） | 7 手动 |
| 4 平衡配重 | |

机器人的轴

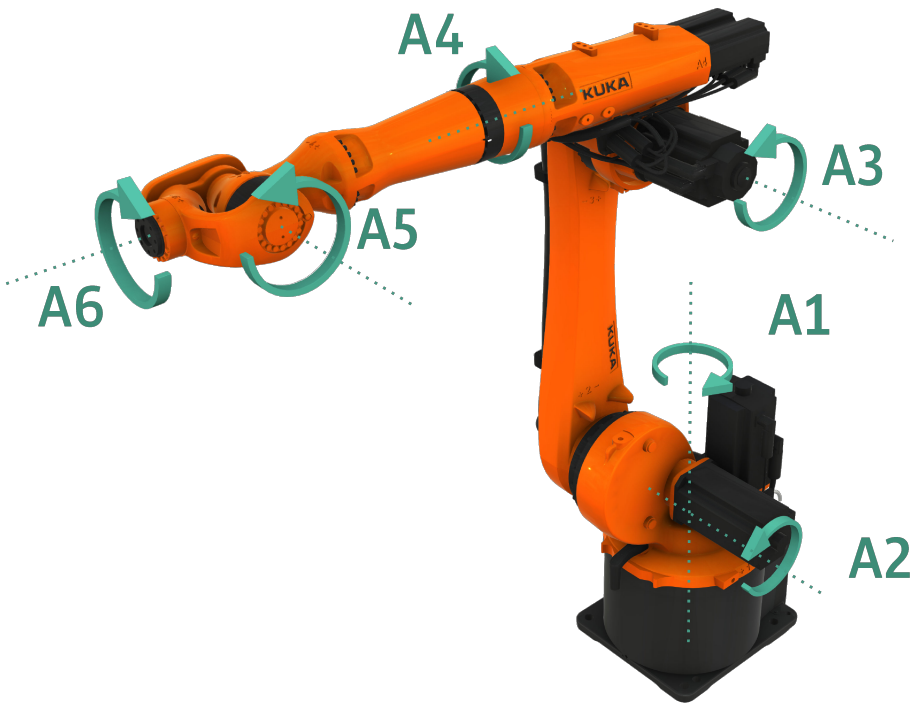


图 1-5：KUKA 机器人的轴

KUKA 产品系列中机械手的技术数据示例：

- 轴数：
 - KR SCARA 有 3 根轴
 - KR40 PA 有 4 根轴

- 在标准垂直折臂机器人时 6 轴
- 7 根轴 用于轻型机器人
- **作用范围:** 从 0.54 m (KR3 R540) 直至 3.9 m (KR 120 R3900 ultra K)
- **自重:** 从 23 kg (LBR iiwa 7 R800) 至 4700 kg (KR1000 Titan)。
- **重复精度:** 0.03 mm 至 0.2 mm 重复精度

机械终端止挡

机器人的基本轴 A1 至 A3 以及手腕轴 A5 的轴范围均由带缓冲器的机械终端止挡限定。
附加轴上可安装其他机械限位。

轴 1	轴 2	轴 3
		

提示

- 如机器人或一个附加轴在行驶中撞到机械终端止挡或轴范围限制处的障碍物或缓冲器，则会导致机器人系统受损。
- 将机器人系统重新投入运行之前，需先联系 KUKA Roboter GmbH 。在继续运行工业机器人之前，被撞到的缓冲器必须立即用新的替换。
- 如机器人（附加轴）以超过 250 mm/s 的速度撞到缓冲器，则必须更换机器人（附加轴）或由 KUKA Roboter GmbH 进行一次重新投入运行。

1.4 机器人控制器 (V)KR C4

谁控制机器人的运动？

机器人机械系统由伺服电机控制运动，而该电机则由 (V)KR C4 控制系统控制。



图 1-6: 控制柜 (V) KR C4

(V)KR-C4 控制系统

机器人控制器（轨迹规划）：控制六根机器人轴以及最多三根附加的外部轴。



图 1-7: (V)KR C4 轴调节器

(V)KR C4 extended 控制系统

机器人控制器（轨迹规划）：控制六个机器人轴以及最多六个附加的外部轴



图 1-8: (V)KR C4 extended 轴调节器

以软件为基础的控制系统

基于 KR C4 的控制系统采用模块化设计方案。**KPC-KUKA PC** 起关键作用。机器人程序根据应用访问所需的硬件接口。可以对**系统相关应用**补充**客户定制应用**。

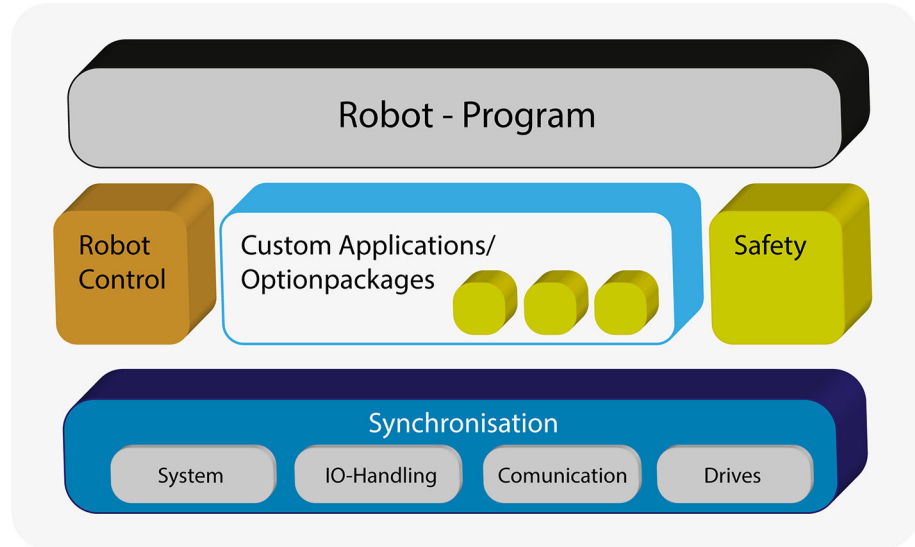


图 1-9：柱状模式

系统相关的应用程序有：

- **Robot Control**
机器人控制的 KUKA 核心系统
- **Safety（安全性）**
KUKA 自己的集成安全控制系统

自定义选项示例：

- **ready2_spot**
用于机器人引导接合/焊接部件
- **ready2_fasten**
用于在工业环境中基于机器人螺接螺栓
- **KUKA.VisionTech**
用于借助一个或多个摄像头确定和纠正相对于部件位置的机器人位置。
- **KUKA.GlueTech**
用于借助涂胶/喷嘴通过机器人引导在一个部件上涂抹胶水。
- **KUKA.Gripper&SpotTech**
用于控制和监控工作环境中的工具和抓爪

通讯方式



图 1-10: (V) KR C4 的通讯途径

- 通过总线系统的通讯方式（例如：ProfiNet、以太网 IP、EtherCAT）
 - 可编程控制器（PLC）
 - 其他控制系统（例如焊接控制系统或机器人控制器）
 - 传感器和驱动器
- 通过网络通讯
 - 主机
 - 笔记本电脑

1.5 KUKA smartPAD

怎样操作 KUKA 机器人？

KUKA 机器人的操作通过示教器（即 KUKA smartPAD）进行。



图 1-11: KUKA smartPAD 型号

KUKA smartPAD 有两种不同型号。

- smartPAD 1
 - smartPAD 2
- smartPAD 2 是 smartPAD 1 的后续版本并且完全向下兼容基于 KR C4 的所有库存控制系统。

KUKA smartPAD 的特点：

- 触摸屏（触敏操作界面）
- 大尺寸竖型显示屏

- KUKA 菜单键
- 可灵活使用的 +/- 键
标准：用作移动键，或调整移动速度
- 操作工艺程序包的按键（如抓爪功能、伺服卡钳、对接）
- 用于程序运行的按键（停止/向前/向后）
- 显示键盘的按键
- 钥匙开关，用于
 - 切换运行方式。
 - 开通其它功能。
- 紧急停止按钮
- 6D 鼠标
- 可热插拔
- USB 接口

1.5.1 smartPAD 1 概览

说明

smartPAD 正面



图 1-12: KUKA smartPAD 前部

序号	说明
1	请求“拔下 smartPAD”的按键
2	选择运行方式的使能开关。开关具有以下规格： • 带钥匙 • 不带钥匙 通过该开关可以调用连接管理器。通过连接管理器可以切换运行模式。
3	紧急停止按键。用于在危险情况下停止机器人。紧急停止按钮在被按下时将自行闭锁。
4	6D 鼠标：用于手动运行机器人
5	点动键：用于手动运行机器人
6	用于设置 POV-程序倍率 的按键
7	用于设置 HOV-手动倍率 的按键
8	主菜单按键：用来在 smartHMI 上显示主菜单。
9	状态键。状态键主要用于设定工艺程序包中的参数。其确切的功能取决于所安装的软件包。
10	启动键：通过启动按键，可启动一个程序。
11	逆向启动键：通过逆向启动按键，可在点动模式下启动一个程序逆向运行。程序将逐步完成。
12	停止键：用停止键可停止运行中的程序。
13	键盘按键： 显示键盘。

smartPAD 背面



图 1-13: smartPAD 背面

序号	元素	说明
1	触笔	用于操作电阻显示屏的无源触控笔
2	使能开关	使能开关具有 3 个位置： • 未按下 • 中间位置 • 完全按下 在 T1 及 T2 运行模式下，有一个使能开关必须保持中间位置，这样才可开动机械手。 在采用自动运行模式和外部自动运行模式时，使能开关不起作用。
3	启动键（绿色）	通过启动按键，可启动一个程序。
4	使能开关	参见 2
5	铭牌	压印的商品号或序列号
6	使能开关	参见 2
7	USB 接口	例如，USB 接口用于存档/恢复。 仅用于 FAT32 格式化的 U 盘。

1.5.2 smartPAD 2 概览

说明

前部



图 1-14：前部

序号	说明
1	2 个有盖子的 USB 2.0 接口 USB 接口可用于例如存档。用于 NTFS 和 FAT32 格式化的 U 盘。
2	准备拔除 smartPAD
3	用于调出连接管理器的钥匙开关。 只有当钥匙插入时，方可转动开关。 利用连接管理器可以预选机器人和工作模式。
4	紧急停止装置 紧急停止按钮。 用于在危险情况下停止机器人。 紧急停止按钮在被按下时将自行闭锁。
5	使用 6D 鼠标可以手动移动机器人。
6	移动键 用运行键手动运行机器人。
7	有尼龙搭扣的手带 如果不使用手带，则手带可能完全被拉入。

序号	说明
8	用于设定程序倍率的按键
9	用于设定手动倍率的按键
10	连接线
11	用户键 用户按键的功能可自由编程设定。用户按键可以用于例如控制外围设备或触发应用特定的动作。
12	启动键 通过启动键可启动一个程序。启动键也可用于手动移至坐标系和将机器人移回到轨迹上。
13	启动退回按键 通过该按键可在程序中反向运行坐标系。
14	停止键 用停止键可停止运行中的程序。
15	键盘按键 显示键盘。
16	主菜单按键 主菜单按键用于在 smartHMI 上显示和隐藏主菜单。

smartPAD 2 背面

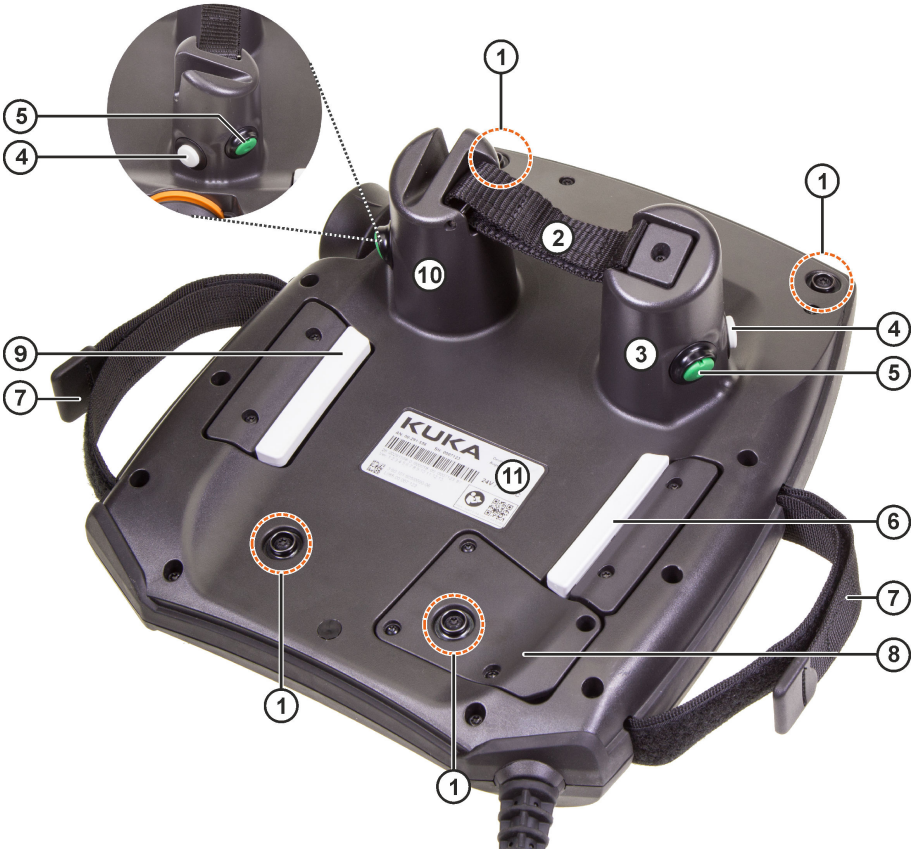


图 1-15: smartPAD 2 背面

序号	说明
1	用于固定（可选）背带的按钮
2	拱顶座支撑带
3	左侧拱顶座：用右手握 smartPAD
4	使能开关 使能开关具有 3 个位置： • 未按下 • 中间位置 • 完全按下（紧急位置） 在 T1、T2 运行模式下，有一个使能开关必须保持中间位置，这样才可开动机械手。 在自动运行模式下，使能开关默认情况下不起作用。
5	启动键（绿色） 通过启动键可启动一个程序。启动键也可用于手动移至坐标系和将机器人移回到轨迹上。
6	使能开关
7	有尼龙搭扣的手带
8	盖板（连接电缆盖板）
9	使能开关
10	右侧拱顶座：用左手握 smartPAD
11	铭牌

1.5.3 插上和拔下 smartPAD

说明

- 可在机器人系统运行时通过请求键取下 smartPAD。可以在任何时间和任何操作方式下拔下 smartPAD。

提示

然而不允许中断正在进行的升级或降级过程。

- 通过插上，smartPAD 将应用机器人控制器当前的运行方式。插上时，smartPAD（固件版本）无关紧要。

提示

可以启动自动升级或降级，不得中断。

- 插入 30 秒后，紧急停止和使能开关方可再次恢复功能。自动重新显示 smartHMI（操作界面）（在 15 秒内）。

提示

在没有事先请求的情况下拔下 smartPAD 将导致机器人紧急停止。

工作原理



警告

- 如果在不事先按下按钮**请求脱开**的情况下拔下 smartPAD，则将触发本地紧急停止。
- 借助按钮触发了请求，但是经过了 25 秒的脱开间隙，然后拔下 smartPAD 时，也同样如此。
- 如果机器人在设备组合体中运行，则可能通过上级安全控制系统触发设备的紧急停止。



警告

- 如果已拔下 smartPAD，则无法再通过 smartPAD 的紧急停止装置来关断设备。
- 因此，在设备上需要一个附加的、无法脱开的外部紧急停止开关。



警告

- 运营商应负责将已拔下的 smartPAD 立即从设备中撤离，并将其妥善保管，使其远离在工业机器人处作业的工作人员的视线和作用范围；或者将其放在隐藏有紧急停止开关的指定支架上。
- 其中目的是为了有效的和无效的紧急停止装置被混淆。



警告

如果没有注意该措施，则有可能造成人员严重身体伤害乃至致命或者巨大的财产损失。



警告

- 将 smartPAD 插到机器人控制器上的操作人员，之后必须留在 smartPAD 旁至少 30 秒。
在该时间之后，至 smartPAD 的连接已恢复。
- 可以确保紧急停止开关和使能开关功能正常。这样就可避免另一用户在紧急情况下使用当前暂时无效的紧急停止。

smartPAD 更新

在启动期间可能会自动更新 smartPAD。通常通过更新控制软件启动。不得中断该更新过程。smartPAD 自己的操作系统将通过控制系统“降级或升级”。



图 1-16: smartPad 更新

提示

中断更新可能导致软件在 smartPAD 上的安装出错。之后 smartPAD 可能无法再启动。

操作步骤

拔下 smartPAD

1. 按下 smartPAD 上的脱离请求键。
在 smartHMI 上显示一条信息并配一个定时器。
计时器会计时 25 秒。在此时间内可从机器人控制器上拔下 smartPAD。

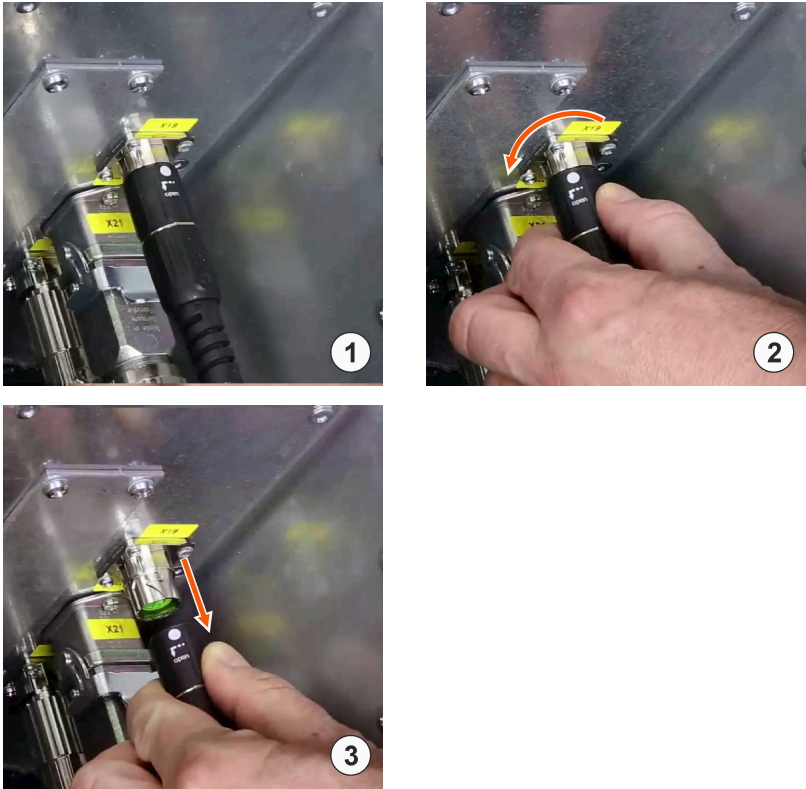


图 1-17: 请求键: 脱离 smartPAD

提示

- 如果在定时器未结束的情况下拔下 smartPAD，会触发紧急停止。
- 只有重新插上 smartPAD 才能取消紧急停止。

2. 从机器人控制器上拔下 smartPAD。



1	插头处于插接状态
2	沿箭头方向将上部的黑色圆环旋转约 25° （见压印）
3	向下拔出插头



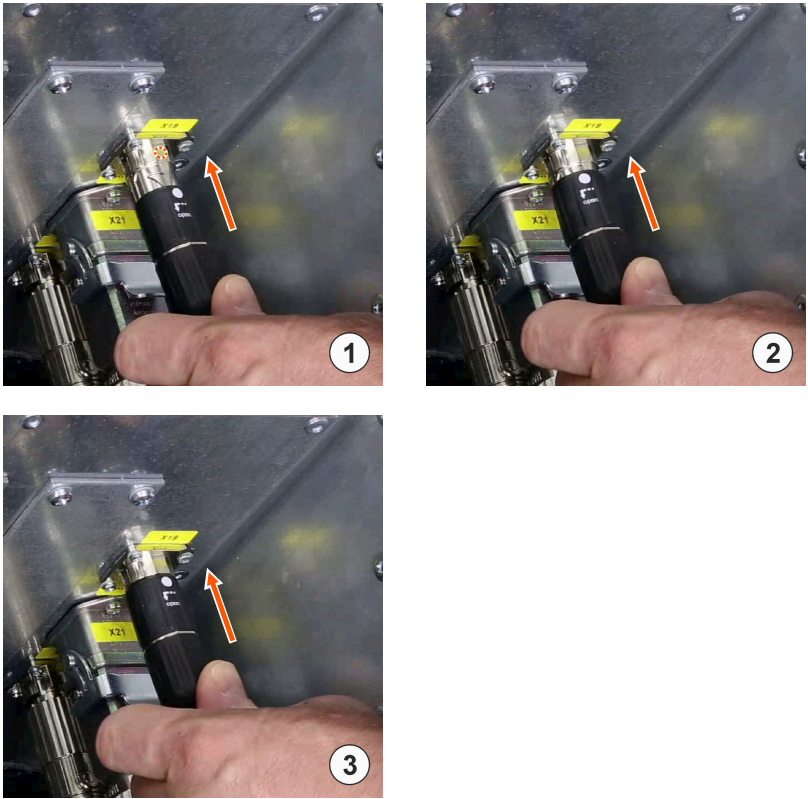
- 如果在计时器计时期期间没有拔下 smartPAD，则此次计时失效。
- 可任意多次按下请求键，以显示定时器。

插上 smartPAD

1. 确保 smartPAD 的本地紧急停止按钮已松开。
2. 将 smartPAD 插头卡在 X19 上。

提示

注意插口和 smartPAD 插头上的标记！



1	插头处于拔下状态（注意标记）
2	向上推插头。在向上推时，上面的黑色圆环自动旋转约 25°。
3	插头自行卡止



警告	
- 将 smartPAD 插在机器人控制器上的用户，之后必须在 smartPAD 旁停留至少 30 秒，直到紧急停止和使能开关重新恢复正常功能为止。 - 这样就可避免另一用户在紧急情况下使用当前暂时无效的紧急停止。	

1.5.4 smartPAD 基本设置

说明

在控制系统启动时以及在连接管理器（>>> 图 1-18）打开时，在 smartPAD 的屏幕上边缘将显示三个绿色符号（1、2 和 3）。这些符号表示当前的控制系统的当前状态以及其与 smartPAD 的连接。在此使用经典的红、黄、绿指示灯颜色。

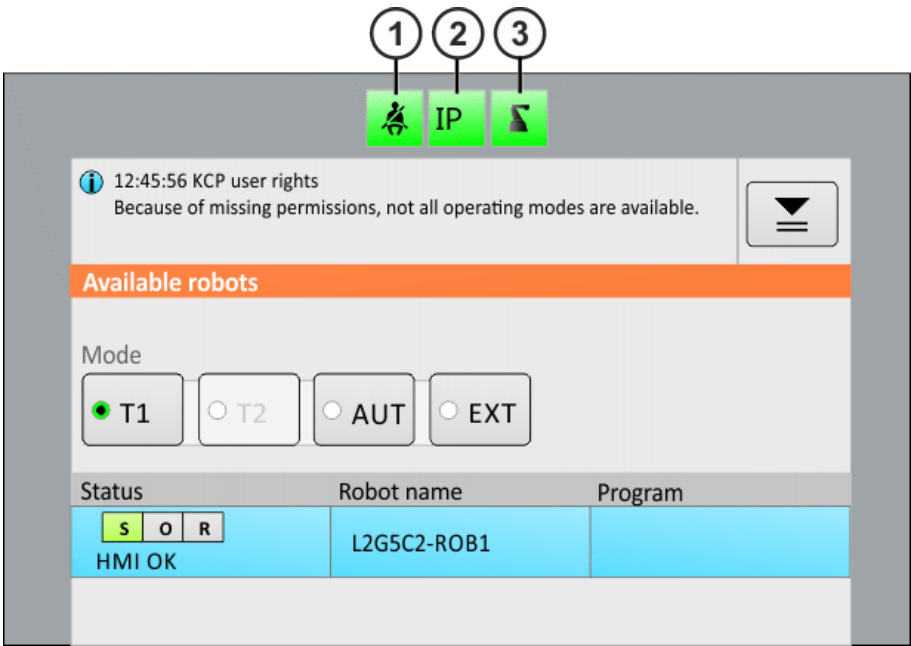


图 1-18：连接管理器

- 1

建立了 Safety 通讯
- 2

IP 地址已成功分配
- 3

KR C4 使用准备就绪

显示 smartPAD 特定的设置/配置

1.

借助钥匙开关打开连接管理器。



图 1-19：连接管理器和运行方式的钥匙开关

2.

按下机器人按键 » 
3.

smartPAD Session Manager 菜单打开。

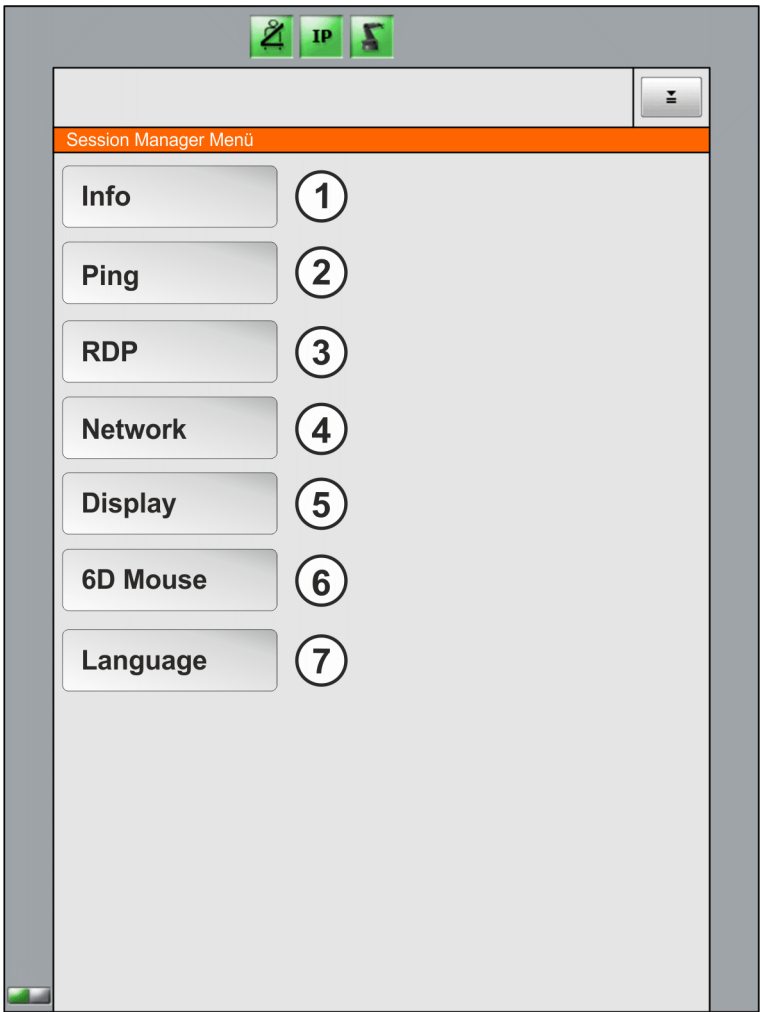


图 1-20：会话（session）管理器菜单

1	信息 版本（smartPAD）概览
2	Ping smartPAD 至 KPC 的连接测试
3	RDP 从 smartPAD 至 KPC 的 IP 和 LOG-IN 数据
4	网络 所用的控制系统网络范围
5	显示 校准和通用屏幕设置
6	6D 鼠标 校准和测试 6D 鼠标
7	语言 语言设置：英语/德语

显示设置

1. 通过显示菜单项（5）打开显示设置。
在该窗口中可调整屏幕亮度以及校准触摸屏。
2. 用**保存**按键（4）应用已更改的设置。

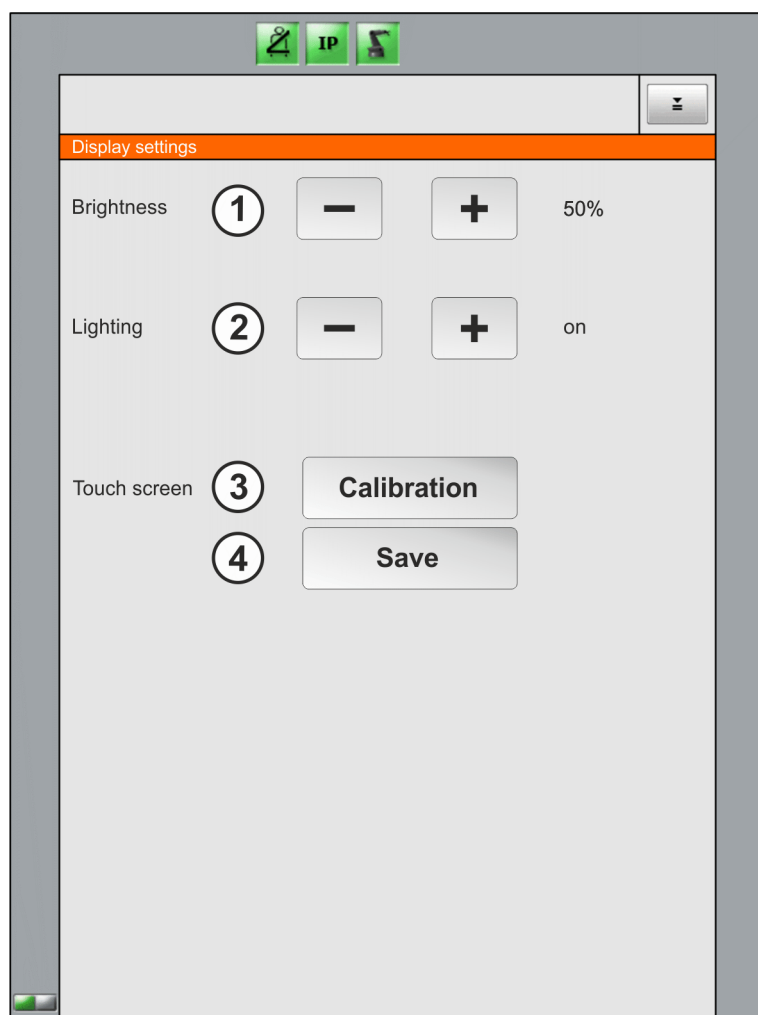


图 1-21: 显示设置

- 1 亮度
- 2 背光
- 3 触摸屏校准
- 4 保存设置

1.6 机器人编程

说明

机器人的编程，其目的是为了重复执行运动语句和流程。

为此，控制器需要大量的信息：

- 当前机器人位置 = 当前工具（Tool）在当前空间（Base）中的位置
- 动作类型
- 速度/加速度
- 等候条件、分支或循环等的信号信息

控制器使用哪种语言？

- 通过预定义好的表单（直接插入表单）创建简单的程序。

示例行指令

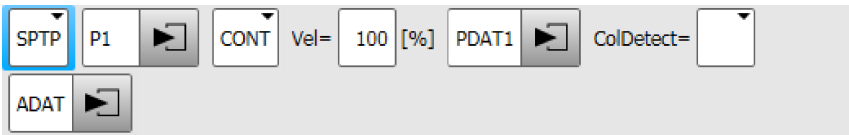


图 1-22：行指令

- 针对循环、逻辑和编程可使用编程语言 **KRL** – **KUKA Robot Language**。
程序举例：

```
LOOP
SPTP P1 Vel=100% PDAT1 Tool[2] Base[4]
SPTP P2 Vel=100% PDAT2 Tool[2] Base[4]
SPTP P3 Vel=100% PDAT3 Tool[2] Base[4]
SLIN P4 Vel=0.5m/s CPDAT4 Tool[2] Base[4]
...
ENDLOOP
```

KUKA 机器人如何编程？

KUKA 机器人可用不同的编程方法编程：

- 以示教（Teach-in）法**在线编程**。



图 1-23：可利用 KUKA smartPAD 进行机器人编程

- **使用 WorkVisual 编程：**用 WorkVisual 软件环境可以创建程序并在线在控制系统上测试。对于故障查找，有调试工具可用。
或者也可以在机器人项目的框架内进行离线编程。

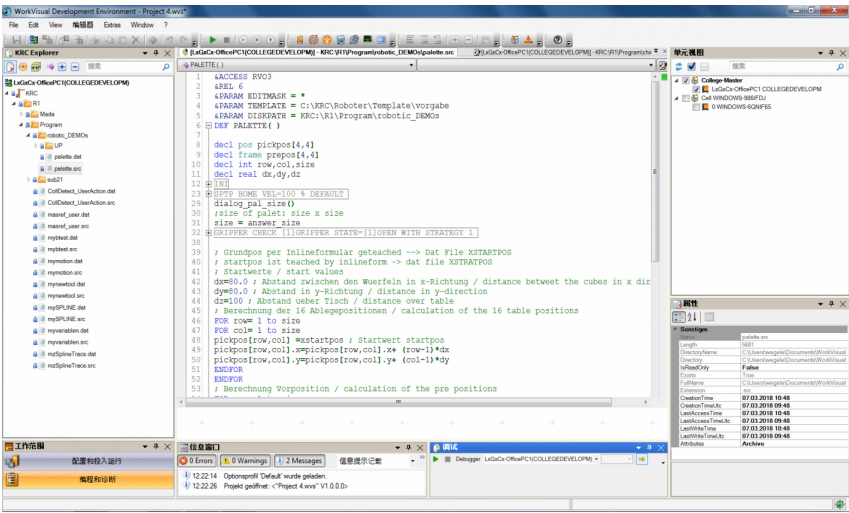


图 1-24: 用 WoV 编程

- 离线编程
 - 图形辅助的互动编程：模拟机器人过程



图 1-25: 用 KUKA Sim 模拟

1.7 机器人安全

示例: College 培训间

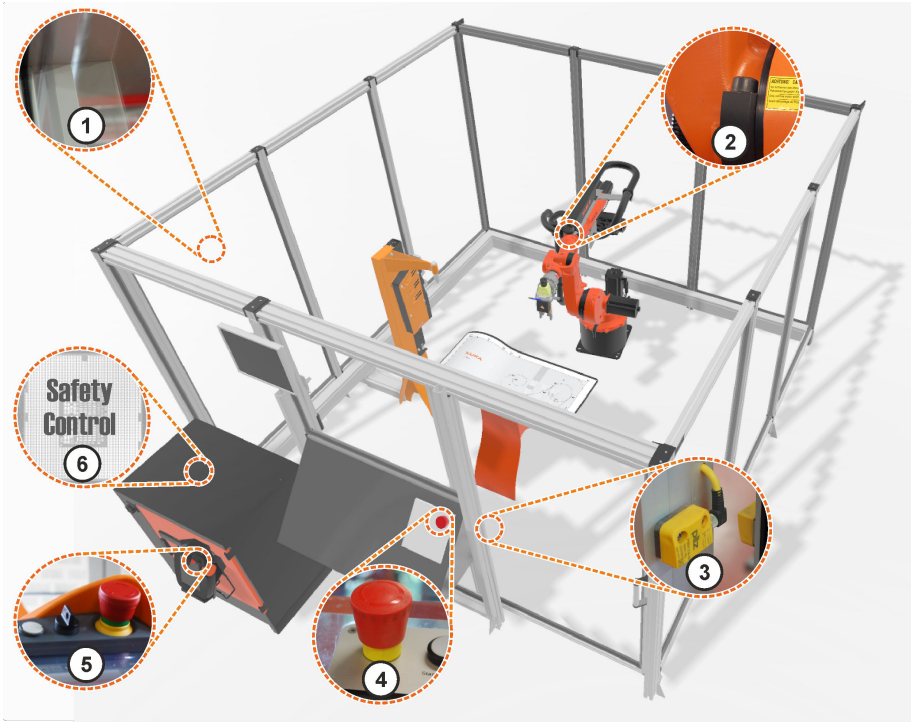


图 1-26: 培训单元

- 1 防护栅栏
- 2 轴 1、2 和 3 的机械终端止挡或者轴范围限制装置
- 3 防护门及用于监控活动防护装置的门触点
- 4 紧急停止装置（外部）
- 5 紧急停止装置、使能开关、调用连接管理器的钥匙开关
- 6 内置的 (V)KR-C4 安全控制器



根据培训间配置的不同，将由一个上级 PLC 识别安全输入端（项号 3 和 4）并将其传输给安全装置的 (V)KR C4。

安全控制器的工作



危险

- 在安全防护装置功能不完善的情况下，机器人系统可能会导致人员或财产受到损失。
- 在安全防护装置被拆下或取消激活的情况下，不允许运行机器人系统。

紧急停止装置

局部紧急停止

- 工业机器人的紧急停止装置是位于 smartPAD 上的紧急停止按钮。
- 在出现危险情况或紧急情况时必须按下此键。

外部紧急停止

- 至少要安装一个外部紧急停止装置，（>>> “外部紧急停止” 页面 32）

- 以确保即使在 smartPAD 已拔出的情况下也有紧急停止装置可供使用。

按下紧急停止按钮时，工业机器人的反应：

- 机械手及附加轴（选配）以安全停止 1 的方式停机。

继续运行

- 若欲继续运行，则必须旋转紧急停止按钮以将其复位，并且对接着出现的信息进行确认。



警告

- 与机械手相连的工具或其他装置如可能引发危险，则必须将其连入设备侧的紧急停止回路中。
- 如果没有遵照执行这一规定，则可能会造成死亡、严重身体伤害或巨大的财产损失。

外部紧急停止

- 在每个可能引发机器人运动或其他可能带来危险情况的工位上都必须有紧急停止装置可供使用。在此由系统集成商来承担责任。
- 至少要安装一个外部紧急停止装置，以确保即使在 smartPAD 已拔出的情况下也有紧急停止装置可供使用。
- 外部紧急停止装置通过客户方的接口连接。外部紧急停止装置不包括在工业机器人的供货范围中。

操作人员防护装置

- **操作人员保护**信号用于锁闭活动的隔离性防护装置，例如防护门。
- 没有此信号，则无法采用自动运行方式。
- 如果在自动运行期间出现信号缺失的情况（例如防护门被打开），则机械手将以安全停止 1 的方式停机。
- 在测试运行模式 T1 及 T2 下，操作人员防护装置不激活。



警告

- 在出现信号缺失后，只有当防护装置已重新关闭且关闭过程得到了确认之后，才可以继续自动运行方式。
- 该确认可以避免在危险区域中有人员停留时因疏忽比如防护门意外闭合而继续进行自动运行。
- 确认必须被设置为可事先对危险区域进行实际检查。
- 其他情况的确认（比如在防护装置关闭时自动确认）是不允许的。
- 系统集成商负责满足这些要求。如果没有达到这些要求，则可能会造成死亡、严重身体伤害或财产损失。

安全运行停止

- 安全停止可通过客户接口上的输入端触发。
- 在安全外部信号为 FALSE 时一直保持该状态。
- 当安全外部信号为 TRUE 时，可以重新移动机械手。
- 不需要确认。

外部安全停止 1 和外部安全停止 2

- 安全停止 1 和安全停止 2 可通过客户接口上的输入端被触发。
- 在安全外部信号为 FALSE 时一直保持该状态。
- 当安全外部信号为 TRUE 时，可以重新移动机械手。

- 不需要确认。

2 机器人系统文档

2.1 学习单元：机器人系统的文件

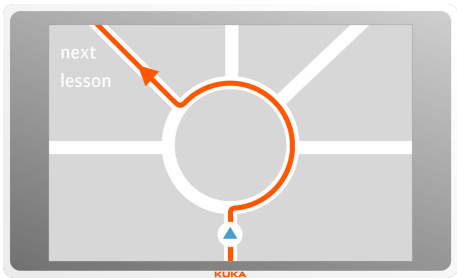


图 2-1：导航器

本学习单元讲授以下内容：

- 介绍 KUKA Xpert
- 注册
- 使用 KUKA Xpert Basic

2.2 KUKA Xpert

说明



图 2-2

KUKA Xpert 是 KUKA 的语义知识库，它包含所有 KUKA 产品的技术信息和文件。基于语义链接的数据，该系统可在回答有关 KUKA 产品的问题时提供支持。

有关主题的所有信息一目了然。通过不同的筛选器可以限制搜索结果。随时可以进行自由文本搜索。



对 KUKA Xpert 网页内容的使用，由用户自行承担责任及风险。免除向 KUKA 提出的任何形式的索赔要求。

许可证

对于 KUKA Xpert，有不同的、可相互组合的许可证软件包可用。下表显示在相应的许可证中包括哪些信息类型：

KUKA Xpert Basic	包括 KUKA 产品的技术文件，如： • 操作和安装说明书 • 操作和编程指南 • 技术说明 • 安全 • 基本产品信息 • 备件信息 KUKA Xpert Basic 已包括在所有其他许可证中。
KUKA Xpert Basic Plus	除了 KUKA Xpert Basic 中的内容之外，还包括客户特有文档的访问。

KUKA Xpert	包括不受限制地访问 KUKA Xpert 的所有内容。这相当于： • 带有超过 20000 条信息连同原因和解决方案的案例数据库 • 超过 500 个工作说明，例如用于更换备件、编程和零点标定 • 超过 300 个用于 KSS 5 - 8.5 的系统变量 • 每台机器人的保养信息，例如：油量、压力和齿形带张力 • 指南，如在总线系统中实现特定功能的步骤说明书 • 功能说明，例如解释总线系统中的功能参数 • 下载，例如代码或配置文件 • 过程力 • 法兰上的负载 • 精度数据 • 兼容性概览
------------	--

2.3 KUKA Xpert URL



只摘要地显示基本许可证。其他许可证模型不是本培训的组成部分。

说明

与 KUKA 产品的随附文件一起交付 KUKA Xpert URL 附页。在其上有通向各个系统组件（机器人/控制系统/软件）文件和其它信息的链接。

KUKA Xpert Basic 含有以下信息：

- 技术资料
- 产品信息
- 备件信息

使用 KUKA Xpert Basic 时需要一个 my.KUKA Login。

KUKA Xpert URL 附页结构如下：

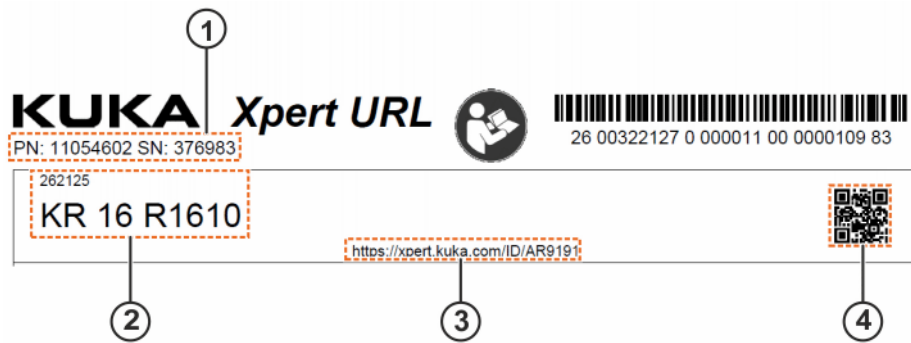


图 2-3：KUKA Xpert URL 附页

序号	说明
1	PN：产品编号 SN：系列号
2	部件编号和产品名称
3	KUKA Xpert URL
4	KUKA Xpert URL 用作二维码

2.4 注册

注册在 my.KUKA Portal 中进行。my.KUKA Portal 是全球 KUKA 客户门户和公司的数字接入点。通过该门户可以找到客户特有的附加信息、数字功能和服务。如果还没有 my.KUKA Login，必须创建一个新账户。



只能用公司电子邮件地址进行注册。不允许用私人电子邮件地址进行注册。

操作步骤

- 1. 输入附页上的链接或扫描二维码。或者在此点击：<https://xpert.kuka.com>。
- 2. 点击您还没有 my.KUKA 账户？，以创建一个新的账户。

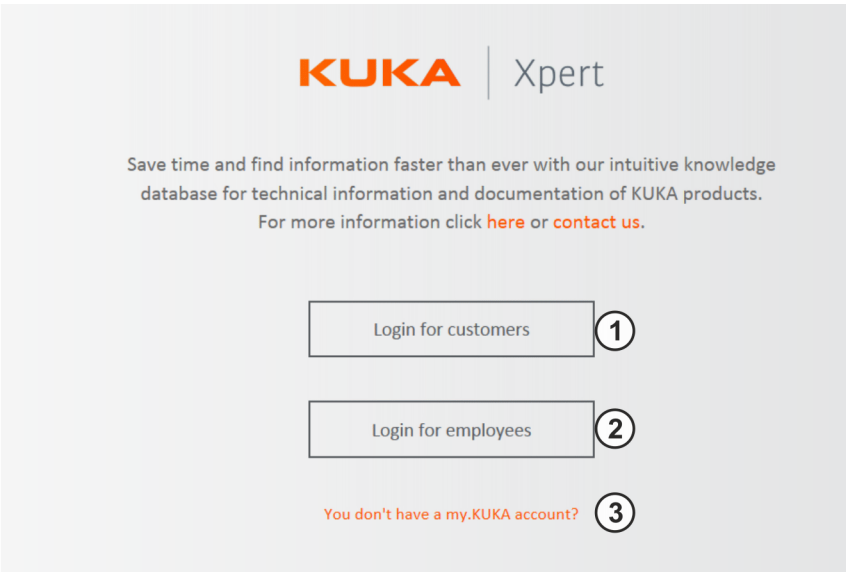


图 2-4

序号	说明
1	客户登录
2	KUKA 员工登录
3	转发到注册表

- 3. 填写注册表并且接受数据保护条件以及使用条件。注册邮件发送到指定的地址。



图 2-5

- 4. 在发送的邮件中点击检查您的账户按钮。网页会自动打开。

- 5. 设置密码并填写其他信息。

2.5 登录

操作步骤

- 1. 输入链接 <https://xpert.kuka.com/Account/Login#/>。
- 2. 点击**客户登录**进行登录。

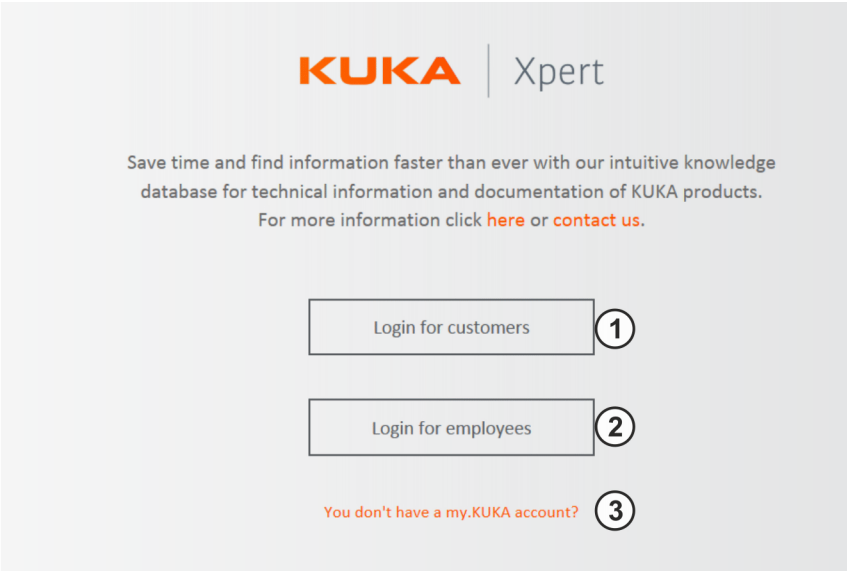


图 2-6

序号	说明
1	客户登录
2	KUKA 员工登录
3	转发到注册表

- 3. 点击 **Login** 或输入电子邮件地址和密码并点击 **Login**。<https://xpert.kuka.com/Account/Login#/>

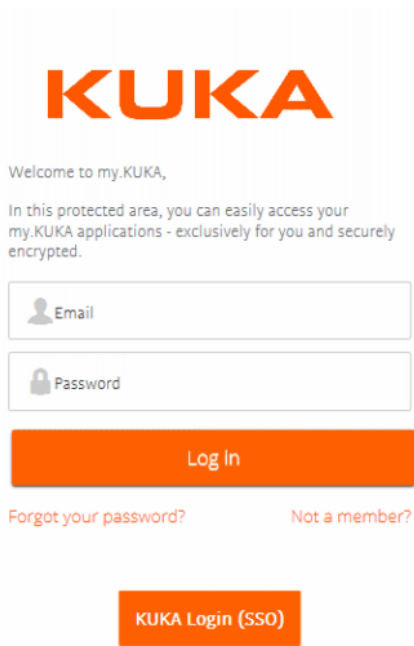


图 2-7

2.6 使用 KUKA Xpert Basic

说明

KUKA Xpert Basic 包含所有技术文件和基本产品信息以及备件信息。

概览

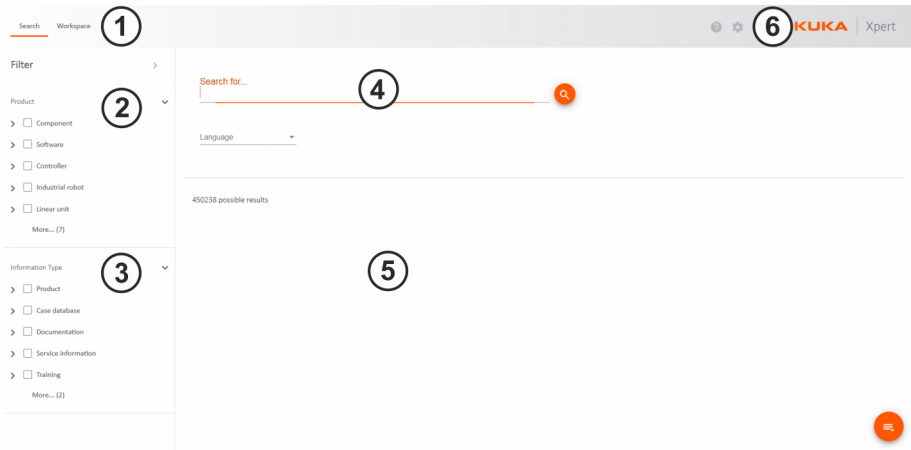


图 2-8: KUKA Xpert 概览

序号	说明
1	概览：确定会话 查找：查找区域 工位：上次查找的历史记录
2	产品：为相关产品设定各个筛选器。无法多选。
3	信息类型：为相关信息类型设定各个筛选器。可以随后筛选查找结果。取决于已选的筛选器或查找结果，在某些情况下可以提供其他的动态筛选器。
4	自由文本查找

序号	说明
5	在查找结果之内的上下文中显示搜索关键词。
6	帮助 设置 退出登录

2.7 练习：使用 KUKA Xpert Basic

练习内容

打开随堂练习本并执行培训师指定的任务。

- 练习：使用 KUKA Xpert Basic

练习内容	
------	--

3 机器人运动

3.1 学习单元：移动机器人

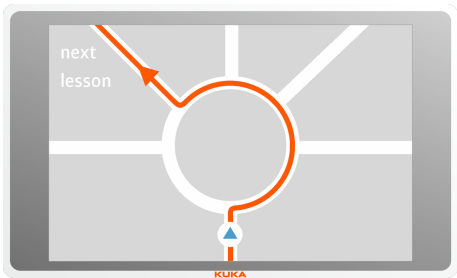


图 3-1：导航器

本学习单元讲授以下内容：

- 信息查看
- 运行方式的选择和设置
- 机器人单轴移动
- 机器人坐标系介绍
- 在世界坐标系中点动机器人
- 借助轨迹运行方式移回并重新向前移动机器人
- 显示机器人位置

3.2 控制器信息读取查看

信息概览



图 3-2：信息窗口和信息提示计数器

1	信息窗口：当前信息的显示
2	信息计数器：每种信息类型的信息数量

控制器与操作员的通信通过信息窗口实现。其中有五种信息类型：
信息类型概览：

图标	类型
	确认信息 - 用于显示需操作员确认才能继续处理机器人程序的状态（例如“确认紧急停止”）。 - 系统输出的确认信息总是导致机器人停止或者不启动。
	状态信息 - 状态信息报告控制系统的当前状态（例如“紧急停止”）。 - 只要这种状态存在，状态信息便无法被确认。
	提示信息 - 提示信息给出正确操作机器人的信息（例如“需要启动键”）。 - 提示信息可被确认。但是不必确认该信息，因为不停止所选的机器人程序或者可以继续进行机器人的手动运行。
	等待信息 - 等待信息说明控制器在等待哪一事件（状态、信号或时间）。 - 等待信息可通过按“模拟”按键手动模拟条件成立。
	对话信息 - 对话信息用作程序控制结构内的询问或用于系统交互。 - 显示带按钮的信息窗口。按钮提供了多种应答方式。



可以被确认的信息可用 **OK** 键确认。用 **全部 OK** 可以一次性确认所有可确认的信息。

提示

按键“模拟”只允许在能够排除碰撞和其他危险的情况下使用！

信息的影响

- 信息会影响机器人的功能。
- 系统输出的确认信息总是导致机器人停止或者不启动。
- 为了使机器人运动，首先必须对信息予以确认。
- 指令“**OK**”（确认）表示请求操作人员有意识地对信息进行分析。



对信息处理的建议：

- 用心地阅读！
- 首先阅读较旧的信息。较新的信息可能是旧信息产生的后果。
- 切勿轻率地按下**全部 OK** 按钮。
- 尤其是在启动后：仔细查看信息。在此过程中让所有信息都显示出来。点按信息窗口即可扩展信息列表。

信息处理

信息中始终包括日期和时间，为追踪相关事件提供准确的时间。

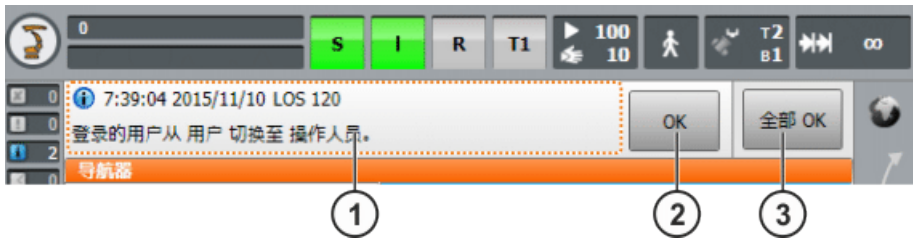


图 3-3：确认信息

观察和确认信息的操作步骤：

1. 点击信息窗口（1），以展开信息提示列表。
2. 确认：
 - 用 **OK**（2）来对各条信息逐条进行确认，
 - 另外一种可选方案：用 **全部 OK**（3）确认所有信息，
3. 再点击一下最上边的一条信息或点击屏幕左侧边缘上的“X”将重新缩小信息列表。

帮助

如果用户对一条信息不理解或需要更详细的解释，则可以调用附加的帮助，这取决于信息类型。

1. 在信息旁点击带问号的按键（1）。

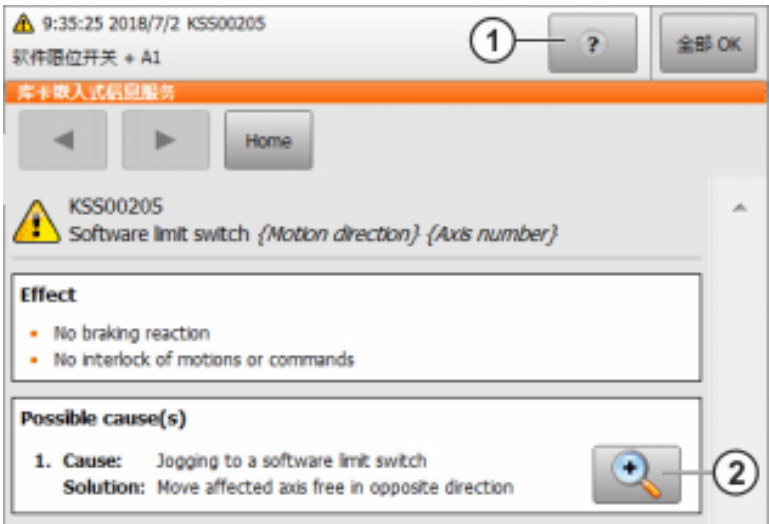


图 3-4：信息，帮助

- 在主窗口中将信息的后果写到程序或机器人系统上。
 - 在可能的原因窗口中将给出一个或多个（可能的）解决方案建议。
 - 其他信息可通过放大镜按键（2）调用。
2. 详细信息页

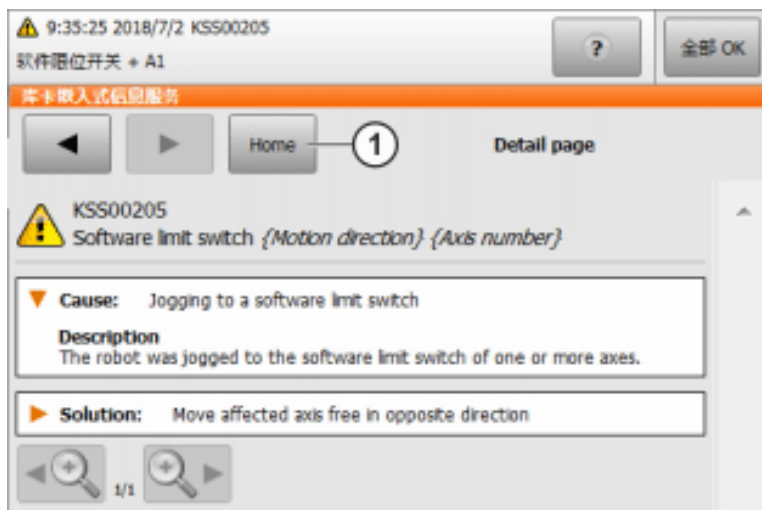


图 3-5：帮助，详情页

- 通过不同的按键（放大镜和箭头）可以在帮助文本内导航。
- 通过 Home 按键（1）可返回到帮助文本的第一页。

3.3 机器人控制系统的运行方式

KUKA 机器人的运行方式



图 3-6：运行方式



相关用户组有哪个运行方式可用的分配可通过管理员用户组进行分配。有时不是所有运行方式都可用。

- **T1**（手动慢速运行）
 - 用于测试运行、编程和示教
 - 程序执行时的最高速度为 250 mm/s
 - 手动运行时的最高速度为 250 mm/s
- **T2**（手动快速运行）
 - 用于测试运行
 - 程序执行时的速度等于编程设定的速度！
用户可借助 POV 程序倍率降低该速度。

- 无法借助 6D 鼠标或运行键移动机器人。
- **AUT** （自动运行）
 - 用于不带上级控制系统的工业机器人
 - 程序执行时的速度等于编程设定的速度！
用户可借助 POV 程序倍率降低该速度。
 - 无法借助 6D 鼠标或运行键移动机器人。
- **AUT EXT** （外部自动运行）
 - 用于带上级控制系统（PLC）的工业机器人
 - 程序执行时的速度等于编程设定的速度！
用户可借助 POV 程序倍率降低该速度。
 - 无法借助 6D 鼠标或运行键移动机器人。

手动慢速运行（T1）

对新的或者经过更改的程序必须始终先在运行方式**手动慢速（T1）**下进行测试。

- 未监控操作人员防护装置（防护门）。
- 应尽可能减少在用防护装置隔离的区域内停留的人数。
如果有多个工作人员在防护装置隔离的区域内停留，则必须注意以下事项：
 - 所有人员必须能够不受妨碍地看到机器人系统。
 - 必须保证所有人员之间都可以直接看到对方。
- 操作人员必须选定一个合适的操作位置，使其可以看到危险区域并避开危险。

手动快速运行（T2）

在**手动快速（T2）**运行模式下：



不过，在 KUKA College 的培训间中与标准情况不同，防护门是受到监控的，必须保持关闭。所有人员应位于由防护装置隔离的区域之外。

- 只有在必须以大于手动慢速运行的速度进行测试时，才允许使用此运行方式。
- 在该运行方式下无法手动运行。
- 通过在 T2 运行模式下运行程序达到编程设定的速度。
该速度可借助 POV 程序倍率降低。
- 机器人操作人员以及其他人员必须处于危险区域之外。

运行方式自动和外部自动

- 必须配备安全、防护装置，而且它们的功能必须正常。
- 所有人员应位于由防护装置隔离出的区域之外。

3.3.1 停止反应

说明

- 工业机器人会在操作或在监控和出现故障信息时做出停机反应。
- 下面的表格中列出了停机反应与所设定的运行方式的关系。

触发器	T1、T2	AUT, AUT EXT
松开启动键	停止 2	-
按下停止键	停止 2	
驱动装置关闭	停止 1	
输入端无“运行许可”	停止 2	
关闭机器人控制系统（断电）	停止 0	
机器人控制系统内与安全无关的部件出现内部故障	停止 0 或停止 1 (取决于故障原因)	
运行期间运行方式切换	安全停止 2	
打开防护门（操作人员防护装置）	-	安全停止 1
松开确认键	安全停止 2	-
将确认键按到底或出现故障	安全停止 1	-
按下紧急停止按钮	安全停止 1	
安全控制系统或安全控制系统外围设备中的故障	安全停止 0	

概念定义

概念	说明
安全运行停止	安全运行停止是一种停机监控。它不停止机器人动作，而是监控机器人轴是否静止。如果机器人轴在安全运行停止时有动作，则安全运行停止触发安全停止 STOP 0。 安全运行停止也可由外部触发。 如果安全运行停止被触发，则机器人控制系统报告安全输出端（X13, SIB Extended）或也可以通过安全的现场总线协议通知上级控制系统。如果在触发安全运行停止时不是所有的轴都停止，并以此触发了安全停止 STOP 0，则也会给该输出端赋值。
安全停止 STOP 0	一种由安全控制系统触发并执行的停止。安全控制系统立即关断驱动装置和制动器的供电电源。 提示：该停止在文件中称作安全停止 0。
安全停止 STOP 1	一种由安全控制系统触发并监控的停止。该制动过程由机器人控制系统中与安全无关的部件执行并由安全控制系统监控。一旦机械手静止下来，安全控制系统就关断驱动装置和制动器的供电电源。 如果安全停止 STOP 1 被触发，则机器人控制系统通过现场总线给一个输出端赋值。 安全停止 STOP 1 也可由外部触发。 提示：该停止在文件中称作安全停止 1。
安全停止 STOP 2	一种由安全控制系统触发并监控的停止。该制动过程由机器人控制系统中与安全无关的部件执行并由安全控制系统监控。驱动装置保持接通状态，制动器则保持松开状态。一旦机械手停止下来，安全运行停止即被触发。 如果安全停止 STOP 2 被触发，则机器人控制系统通过现场总线给一个输出端赋值。 安全停止 STOP 2 也可由外部触发。 提示：该停止在文件中称作安全停止 2。

概念	说明
停机类别 0	驱动系统立即关闭，制动器制动。机械手和附加轴（选项）在额定位置附近制动。 提示： 此停机类别在文件中被称为 STOP 0。
停机类别 1	1 秒钟后驱动装置关断，制动器制动。 机械手和附加轴（选项）顺沿轨迹制动。 提示： 此停机类别在文件中被称为 STOP 1。
停机类别 2	驱动系统不关闭，制动器不制动。机械手及附加轴（选项）以沿轨迹的制动斜坡进行制动。 提示： 此停机类别在文件中被称为 STOP 2。

3.3.2 切换运行模式

操作步骤



如果在运行过程中改变了运行方式，驱动装置即立刻关断。工业机器人以安全停止 2 停机。

1. 在 smartPAD 上转动用于连接管理器的开关。连接管理器随即显示。



2. 选择运行模式。



图 3-7：运行模式

- 通过点击相关的按键选择所需的运行方式（T1，T2，AUT，EXT）。
 - 所选的运行模式用绿点标记。
3. 将用于连接管理器的开关再次转回初始位置。

所选的运行方式会显示在 smartPAD 的状态栏中。



- 

在 VSS - 大众系统软件中**没有**运行方式 AUT 可用。
- 

KSS 8.5 以上，通过权限管理可以有针对性地为相应的用户组分配和禁用运行模式。

3.4 机器人单轴移动

单轴移动机器人轴

机器人的每根轴都可以单独沿正向和负向运动。使用点动键或者 KUKA smartPAD 的 6D 鼠标。可用 HOV 手动倍率降低移动速度。仅在 T1 运行模式下才能手动运行，为此必须将一个确认键在中间位置按住。

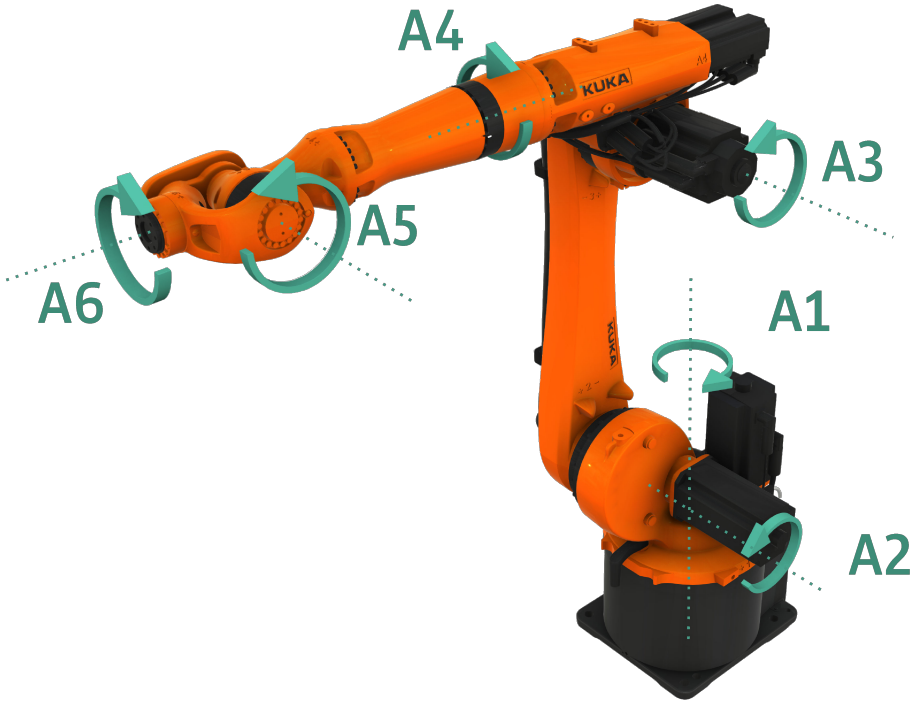


图 3-8: KUKA 机器人的轴

- 

图中箭头分别指向正的移动方向 (+)。

原理

- 通过按下一个确认键激活驱动装置。如果驱动装置许可开通，则点动键的文字说明为绿色。只要一按点动键或 6D 鼠标，机器人轴的伺服电机便启动，机器人执行所需的运动。
- 运动可是连续的，也可是步进式的。为此要在状态栏中选择步距（增量）。（>>> “操作步骤” 页面 53）

以下信息对手动运行有影响：

信息	原因	补救措施
“激活的指令被禁”	出现（停机）信息或引起激活的指令被禁的状态（例如：按下了紧急停止按钮或驱动装置尚未就绪）。	解锁紧急停止按钮并且在信息窗口中确认信息。按下一个使能开关后可使用驱动装置。
“软件限位开关-A5”	沿给定的方向（+ 或 -）移到所显示轴（例如 A5）的软件限位开关。	显示的轴朝相反方向移动。

实用提示

1. 选择所需的程序。
2. 按下一个使能开关。
3. 等到驱动装置准备就绪（移动键亮绿色，驱动装置 I 显示绿色）。
4. 然后开始所需的操作 » 启动键、移动键或 6D 鼠标。



如果不遵守该顺序，则显示信息“激活的指令已锁闭”。

与单轴点动的安全提示

运行模式

- 机器人只有在运行方式 T1 （手动降低的速度）下才可手动运行。
- 在 T1 模式下，手动运行速度最大为 250 mm/s。
- 通过连接管理器设置运行方式。

使能开关

- 为了能移动机器人，必须按下使能开关并按住。
- smartPAD 上装有三个使能开关，smartPAD2 上则有四个。
- 使能开关具有 3 个挡位：
 - 未按下
 - 中间位置
 - 完全按下（紧急位置）

软件限位开关

- 机器人的移动受到软件限位开关位开关的最大正、负值的限制。

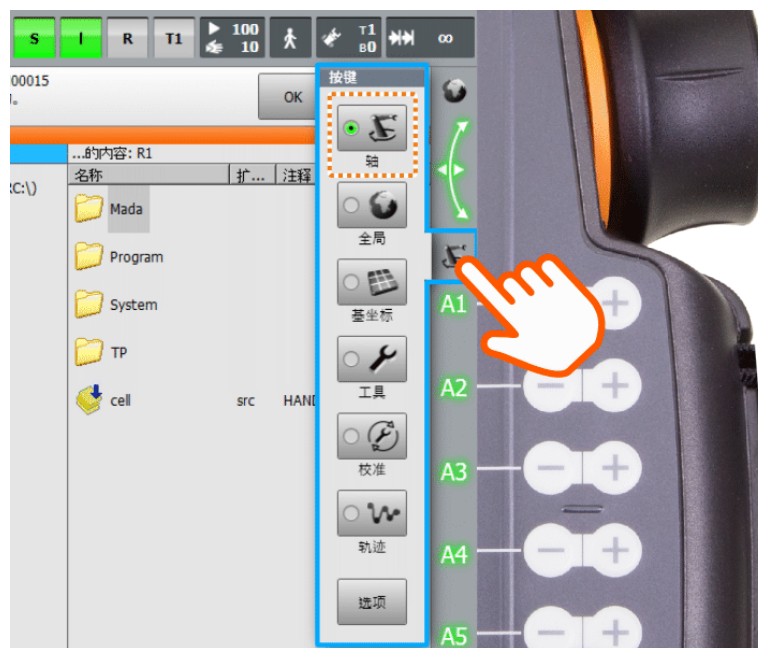
提示

- 如果在信息窗口中出现信息“执行零点标定”，则也可超过这两个极限值移动。
- 但这可能会损坏机器人系统！

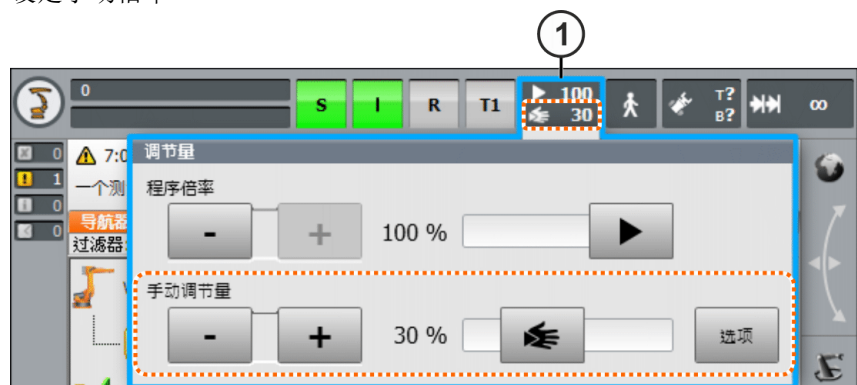
操作步骤

执行与轴相关的运动

1. 选择轴作为移动键的选项



2. 设定手动倍率



3. 将确认开关按至中间挡位并按住



4. 在运行键旁边将显示轴 A1 至 A6。
按下正向或负向移动键，以使轴朝正向或负向运动。



3. 4. 1 增量式手动运行

说明

增量式手动运行

- 增量式点动运行模式可以使机器人移动所定义的距离，如 10 mm 或 3°。
- 运行时可以用点动键接通增量式点动运行模式。
用 6D 鼠标运行时不能用增量式手动运行模式。
- 相应的移动键必须保持按下，直到达到预设的距离为止。达到预设的距离时机器人自动停止。

应用范围:

- 以同等间距进行点的定位
- 如果距离已知（单位为 mm 或度），例如测量装置中的数据，则可以进行精确的点修正
- 从一个位置驶出所定义距离，如在故障情况下

前提条件

- 运行模式 T1

操作步骤

1. 在状态栏中选择增量值 (1):



图 3-9：增量式手动运行

2. 使用移动键可以采用笛卡尔或轴坐标方式运行机器人。



如果机器人的运动因松开使能开关等被中断，则在下一个运动中，中断的行驶距离不会继续，而是重新开始。

设置/增量值：

设置	说明
连续	已关闭增量式手动移动。
100mm / 10°	1 增量 = 100 mm 或 10°
10mm / 3°	1 增量 = 10 mm 或 3°
1mm / 1°	1 增量 = 1 mm 或 1°
0.1mm / 0.005°	1 增量 = 0.1 mm 或 0.005°

增量单位为 mm：

- 适用于在 X、Y 或 Z 方向进行的笛卡尔运动。

增量单位为度：

- 适用与在 A 、B 或 C 旋转方向进行的笛卡尔运动。
- 适用于与轴相关的运动中。

3.4.2 在紧急情况下不用控制系统移动机器人

说明

发生事故或故障后，可借助自由旋转装置移动机器人。
自由旋转装置可用于主轴驱动电机，根据机器人类型的不同，有些也可用于中心轴电机。
它只允许用于特殊情况和紧急情况，例如用于解救被困人员。



图 3-10: 自由旋转装置

如果已经使用了自由旋转装置，则必须确保制动器功能正常。

- 为此需要执行制动测试。如果没有通过此测试，则需要更换相应的电机。
- 如果不能提供控制系统的制动测试或无法进行测试，则更换相应的电机。



制动测试是 KSS8.x 的固定功能。但是，该功能必须由调试员/安全调试员开通和激活。

操作步骤



警告

- 运行期间，电机将达到的温度可导致皮肤烧伤。
- 避免与其接触。
- 请务必采取适宜的安全防护措施，例如佩戴合适的防护手套。

1. 关断机器人控制器，并锁住（例如用挂锁），防止未经许可的意外重启。
2. 拆下电机处的防护盖。
3. 将自由旋转装置置于相应的电机上，并将轴朝所希望的方向运动。



- 可作为选项订购在电机上用箭头标记方向的标签。
- 须克服电机机械制动器的阻力，且必要时还须克服额外的轴负载。

轴 2 电机上的示例：

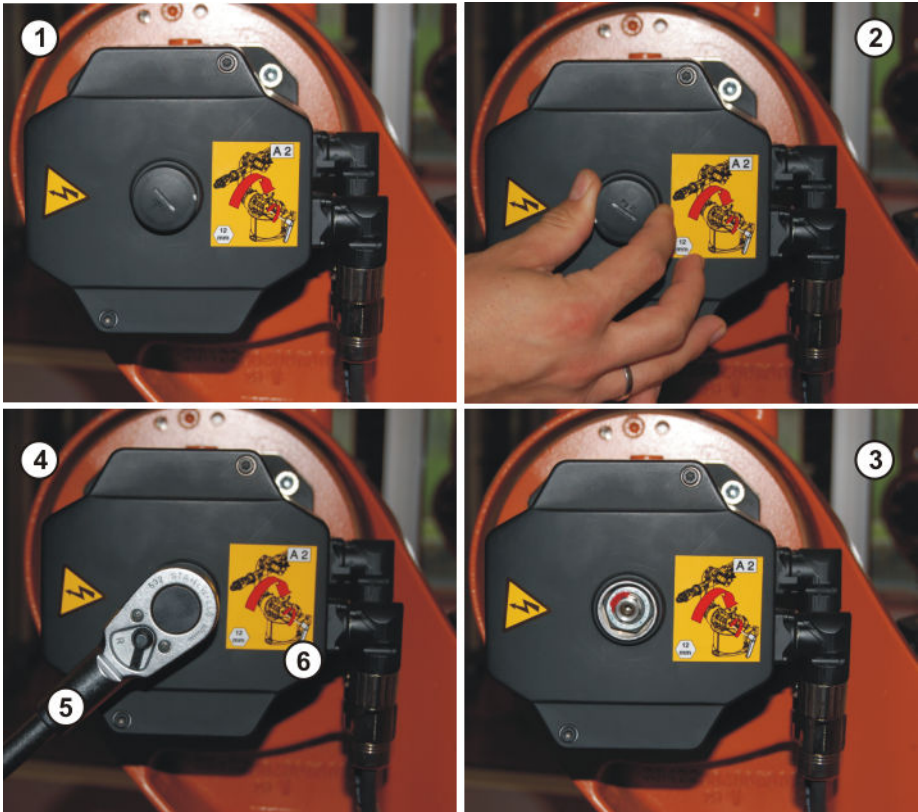


图 3-11：使用自由旋转装置的操作步骤

序号	说明
1	防护盖盖上的电机 A2
2	打开电机 A2 的防护盖
3	防护盖已拆下的电机 A2
4	将自由旋转装置装到电机 A2 上
5	自由旋转装置
6	有转向说明的标签（选项）



小心

- 在使用自由旋转装置移动轴时，可能会损坏电机制动器。
- 可能会导致人员伤害及设备损伤。
- 在使用自由旋转装置后，必须进行制动测试或更换相应的电机。



更多信息请见 KUKA 机器人的操作和安装指南。

3.4.3 练习：操作及轴相关的手动运行

练习内容

打开课程练习本并完成以下练习：

- 操作以及与轴相关的手动运行

3.5 与机器人相关的坐标系

在工业机器人的操作、编程和投入运行时坐标系具有重要的意义。在机器人控制器中定义了下列坐标系：

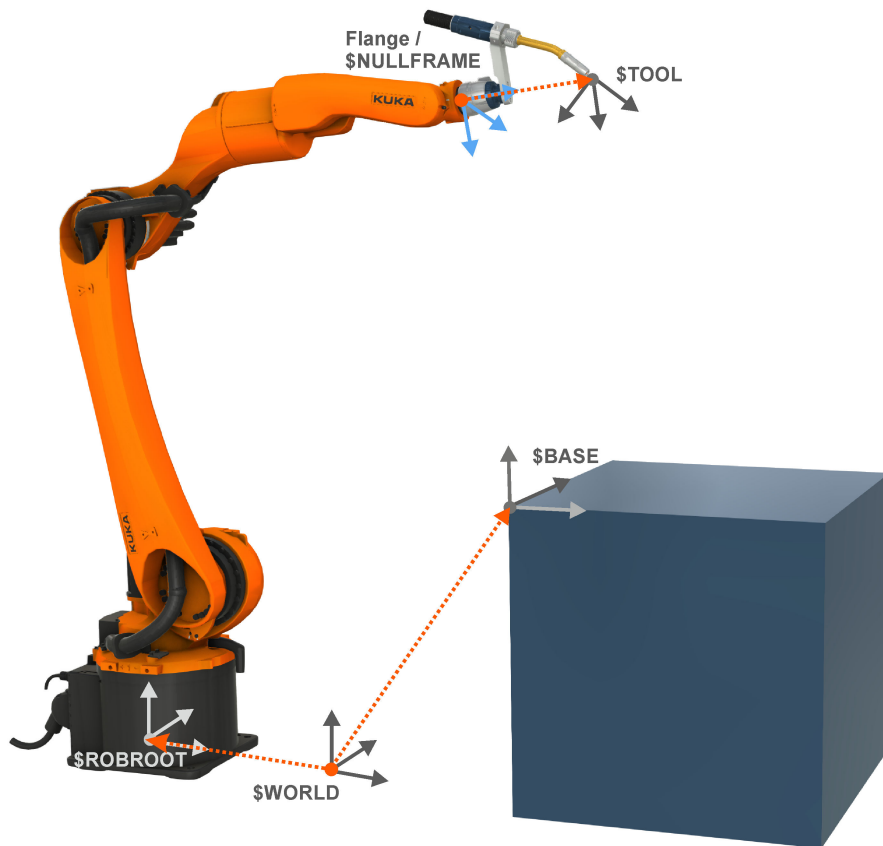


图 3-12: 坐标系统一览

- **ROBROOT**
机器人根部坐标系
 - 固定位于机器人足部。
 - 是机器人的原点。
 - 可以参照 WORLD 坐标系移动 ROBROOT。
- **WORLD**
世界坐标系
 - 在供货状态下与 ROBROOT 坐标系中一致。
 - 可以从机器人根部“向外移出”。
 - 定义世界坐标系在 ROBROOT 坐标系中的位置。
 - 此外，机器人系统为壁装与吊顶安装时使用。
- **BASE**
基坐标系
 - 是一个可自由定义、用户定制的坐标系
 - 定义基坐标在世界坐标系中的相对位置。
- **FLANGE / NULLFRAME**
法兰坐标系
 - 法兰坐标系固定位于机器人法兰上。
 - 原点为机器人法兰中心。
 - 是工具坐标系的参照。

- 还被称为 NULLFRAME。
- **TOOL**
工具坐标系
 - 是一个可自由定义的坐标系。
 - TOOL 坐标系的原点被称为 **TCP** (Tool Center Point) 即工具中心点。
 - 用于测量工具。

3.5.1 坐标系和“右手法则”

前面几节已经介绍了机器人系统上的坐标系作用的位置与方式。但在日常生活中常常会有这样的问题：“我要在 smartPAD 上按哪个键，才能使机器人沿特定方向移动呢？”我们一直拥有一个辅助工具，便于我们生动理解，那就是自己的右手。

姿态

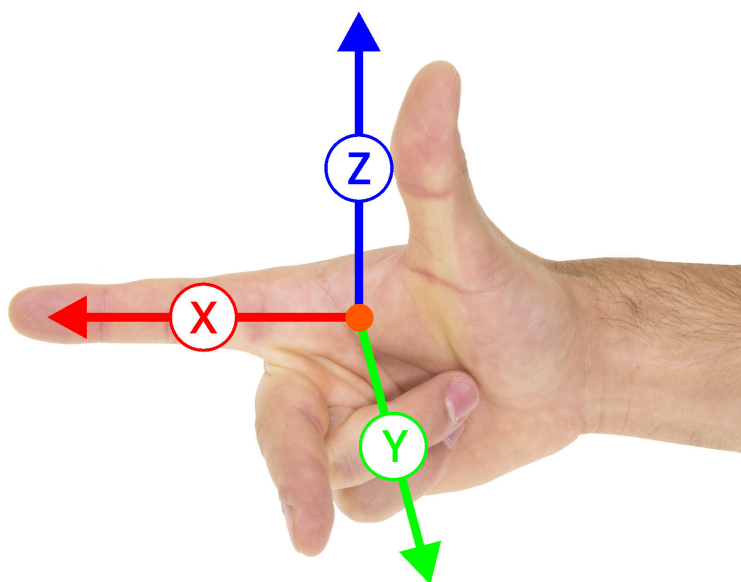


图 3-13: 姿态，右手

将大拇指、食指和中指摆成一个直角坐标系，例如使食指与 X 轴、大拇指与 Z 轴重合。现在中指指向 Y 轴方向。指尖分别指向 (+) 正的移动方向。

转动

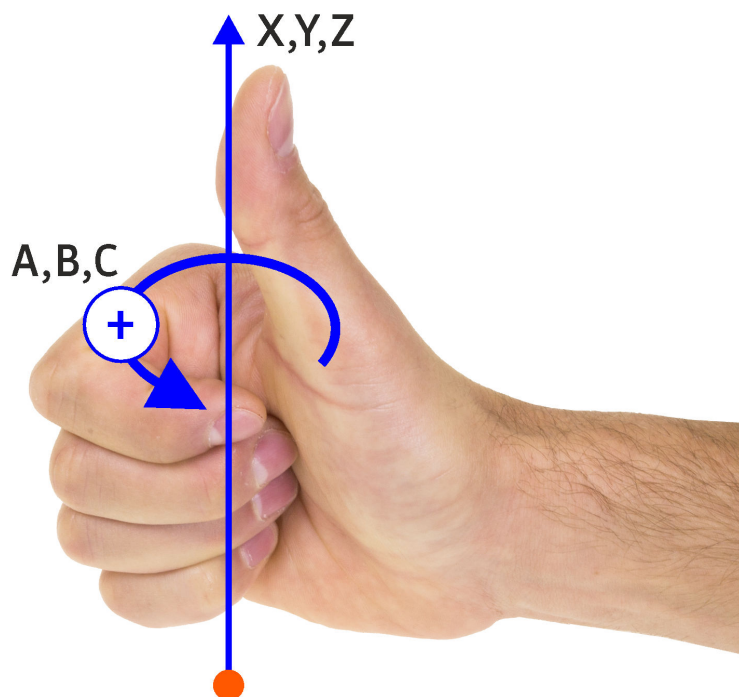


图 3-14: 旋转, 右手

将右手团成拳, “虚拟地”握住所需轴。伸出的大拇指指向“正”向。其他所有手指则指向正的旋转方向。

3.6 在世界坐标系中点动机器人

世界坐标系中的运动

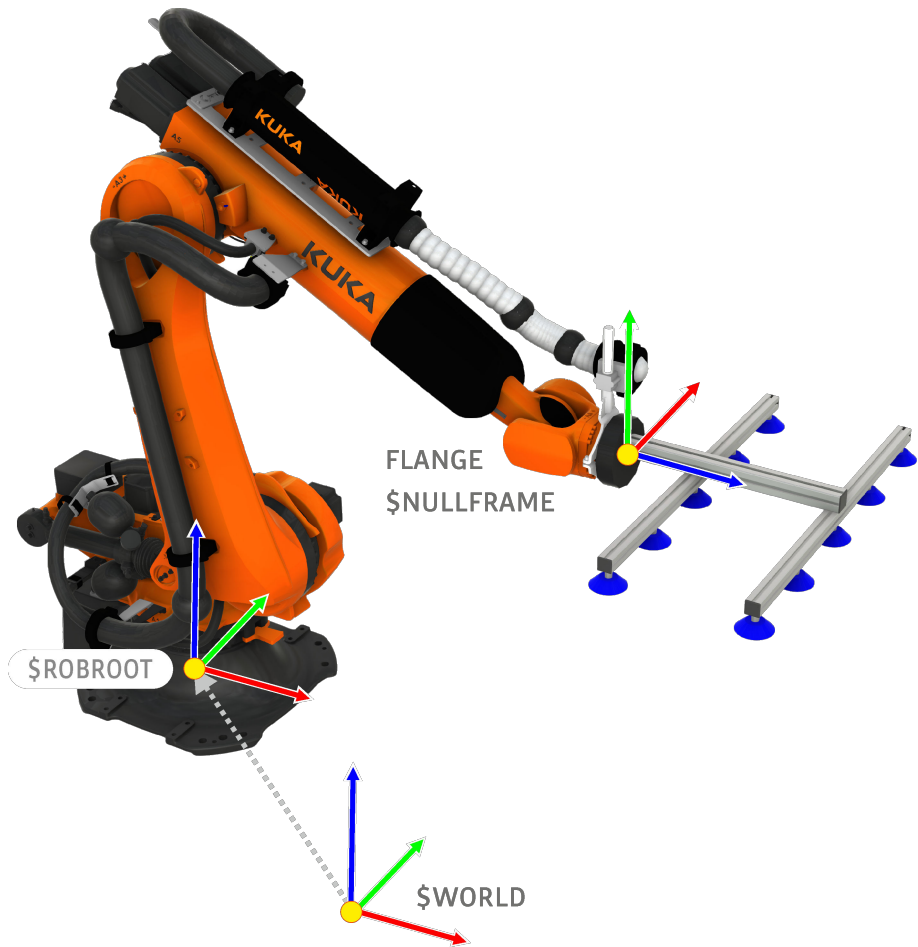


图 3-15：世界坐标系点动运行的原则

- 法兰中心点（零坐标系）可根据世界坐标系的坐标方向移动。同时，**所有**机器人轴都移动。
- 为此需要使用点动键或者 KUKA smartPAD 的 6D 鼠标。
- 在默认设置下，世界坐标系位于机器人根部（\$ROBROOT）中。
- 速度可以更改（手动倍率：HOV）
- 仅在 T1 运行模式下才能手动运行。
- 使能开关必须保持在中间位置上。

6D 鼠标

- 通过 6D 鼠标可以使机器人的运动变得直观明了，因此是在世界坐标系中进行点动运行时的不二之选。
- 鼠标位置和自由度两者均可更改。

在世界坐标系中手动运行的方式

在坐标系中可以两种不同的方式移动机器人：

- 沿坐标系的坐标轴方向**平移**（直线）：X、Y 和 Z。
- 环绕着坐标系的坐标轴方向**转动**（旋转/回转）：角度 A、B 和 C。

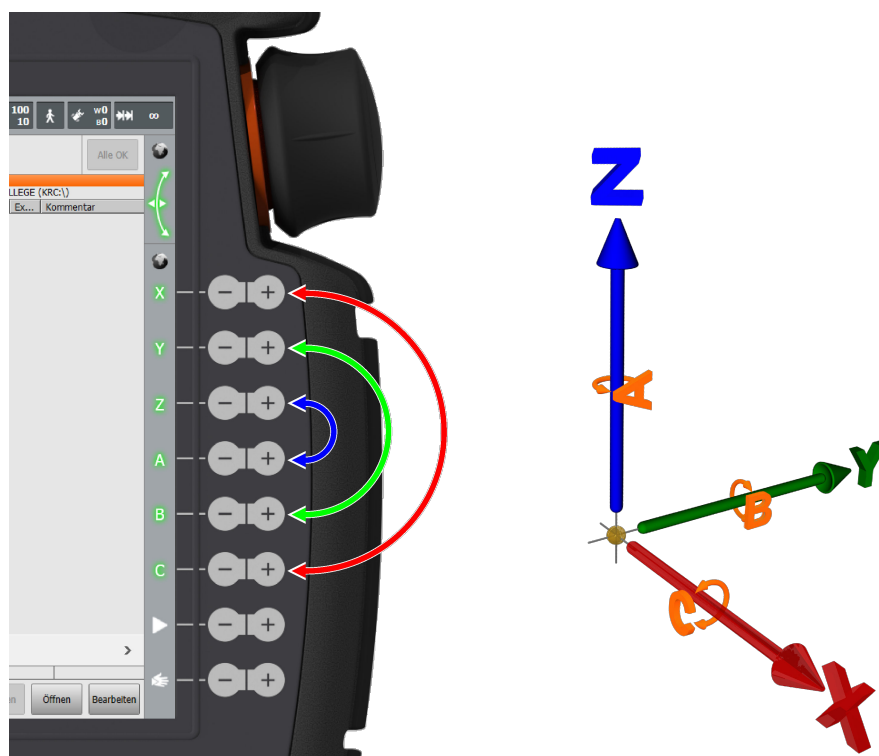


图 3-16：笛卡尔坐标系

收到运行指令时（例如按了移动键后）控制系统会先计算一行程段。该行程段的起点是工具参照点（TCP - Tool Center Point）。行程段的方向由世界坐标系给定。控制系统控制所有轴的运动，使工具沿该行程段运动（平移）和绕其旋转（转动）。

优点

- 机器人的运动始终可预测。
- 空间中的工具中心点的运动始终是唯一的，因为原点和坐标方向始终是已知的。
- 对于经过零点标定的机器人始终可用世界坐标系。
- 可用 6D 鼠标直观操作。

缺点

- 机器人可能会移至奇点位置。
- 在这种情况下，必须借助轴相关的运行方式使机器人从奇点中驶出。

6D 鼠标的使用

可将 6D 鼠标用于所有运动方式：

- **平移：**通过按下和拖动 6D 鼠标

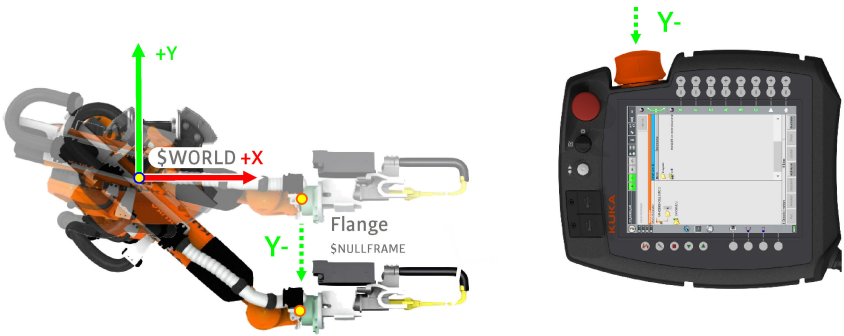


图 3-17：示例：向左运动

- **转动：**通过转动并摆动 6D 鼠标

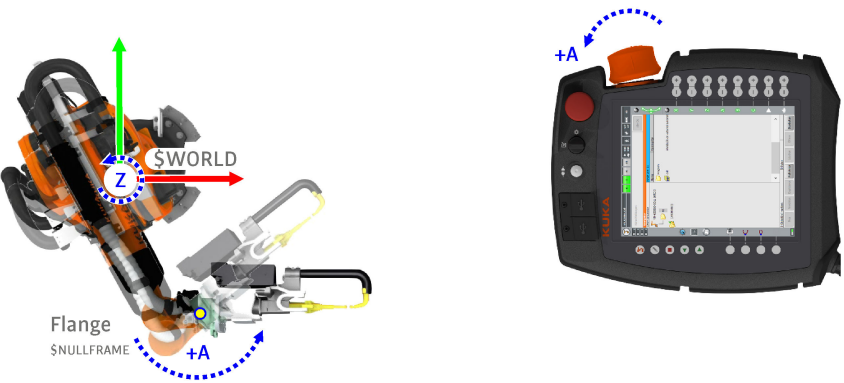


图 3-18：示例：绕 Z 轴的旋转运动：角度 A

操作员位置

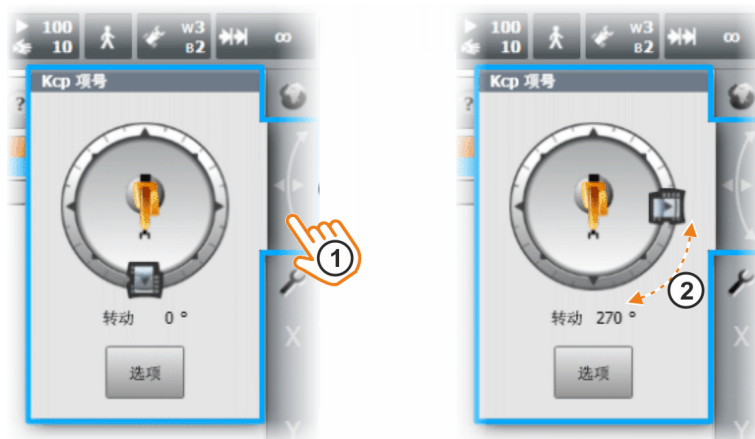
可根据操作员相对于机器人的位置调整 6D 鼠标的位置。



图 3-19: 6D 鼠标, 位置

执行平移（世界坐标系）

1. 调整 smartPAD 位置。



KCP-KUKA 控制面板 $\triangle$ smartPAD

- 通过点击 6D 鼠标旁的箭头 (1) 打开选项窗口。
 - 通过移动 smartPAD 图标更改 smartPAD 位置。
2. 其他鼠标设置可通过**选项**按钮调整。



图 3-20：手动移动选项

- 1 鼠标设置
6D 鼠标自由度的限制/扩展
- 2 同步坐标系
坐标系设置适用于鼠标和移动键
- 3 坐标系
要在其中用 6D 鼠标移动机器人的坐标系



点击 SYNC 按键可以同步 6D 鼠标和移动键的坐标系选择。

- 3. 将世界 (1) 选为 6D 鼠标的选项

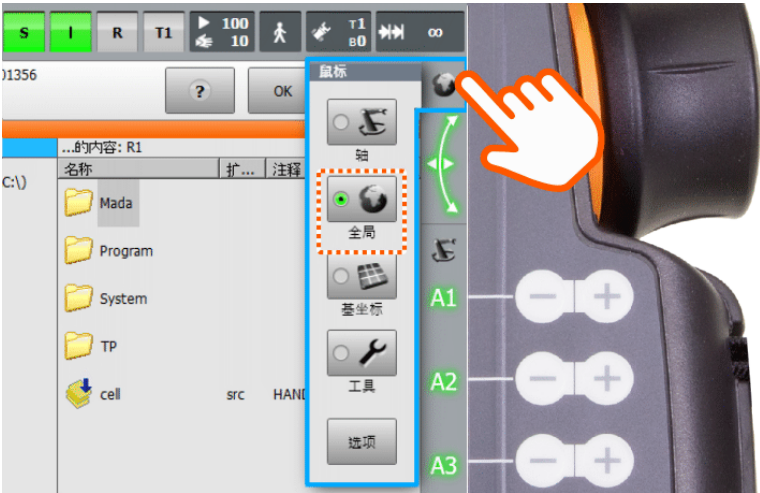
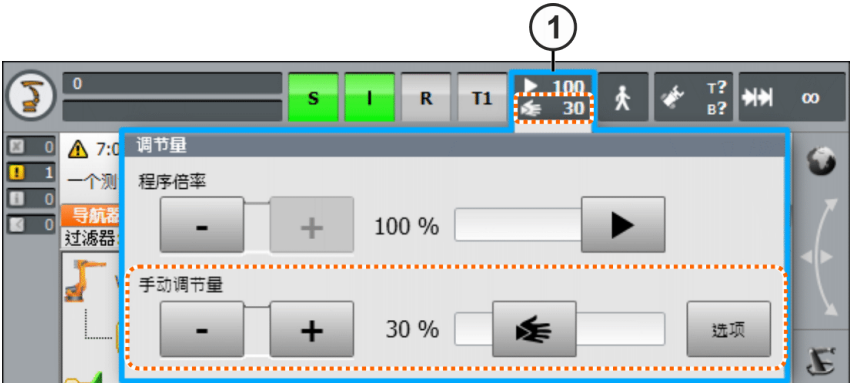


图 3-21：在世界坐标系中点动运行，6D 鼠标

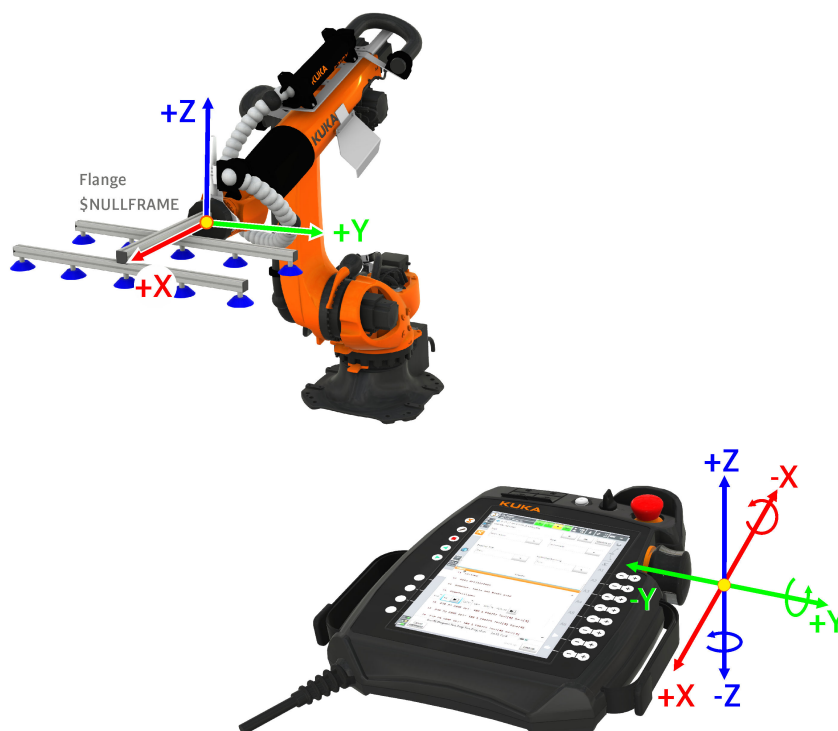
- 4. 设定手动倍率 (1)



5. 将四个使能开关中的一个按至中间位置并按住。



6. 用 6D 鼠标将机器人朝所需方向移动。



7. 此外也可使用点动键。



图 3-22：在世界坐标系中点动运行，点动键

3.6.1 练习：借助移动键在世界坐标系中操作以及手动运行

练习内容

打开课程练习本并完成以下练习：

- 借助运行键在世界坐标系中操作以及手动运行

3.6.2 练习：借助 6D 鼠标在世界坐标系中操作以及手动运行

练习内容

打开课程练习本并完成以下练习：

- 借助 6D 鼠标在世界坐标系中操作以及手动运行

3.7 借助轨迹运行方式移动机器人

说明

如果机器人运动，则机器人控制器周期性地在 一个缓冲器中记录当前机器人位置。这不取决于机器人如何运动。即使在程序调用和不时的手动运行之间切换，也不会对位置记录有影响。可以反向运行至机器人刚刚过去的状态。机器人可以在任意位置上借助**轨迹**运行方式和 **±** 键移动，也可重新返回至初始位置。

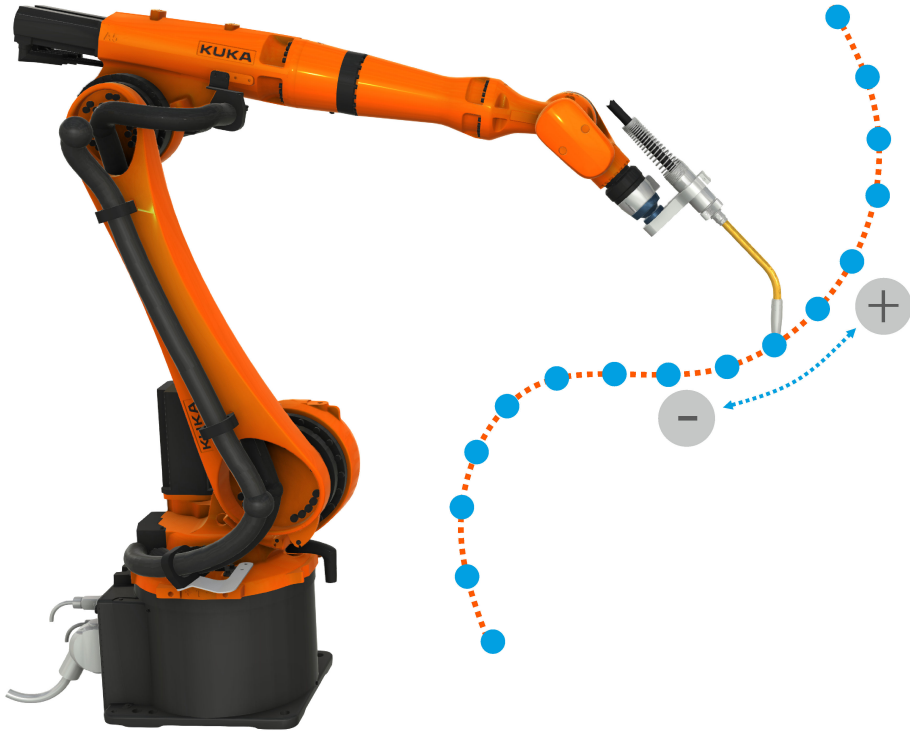


图 3-23：用轨迹方式运行机器人

-  执行的轨迹接近原始轨迹。该轨迹可通过借助样条轮廓连接保存的点确定。
-  通过点动键反向运行的缓冲器与通过 smartPAD 启动反向键反向运行的缓冲器不同。
-  如果在运行时通过软件限位开关限定机器人，则复位该记录并开始新的记录。

操作步骤

-  只在运行方式 T1 下使用轨迹移动方式。

- 在窗口 **手动移动选项** 的选项卡 **按键** 中选择选项 **轨迹** (1)。

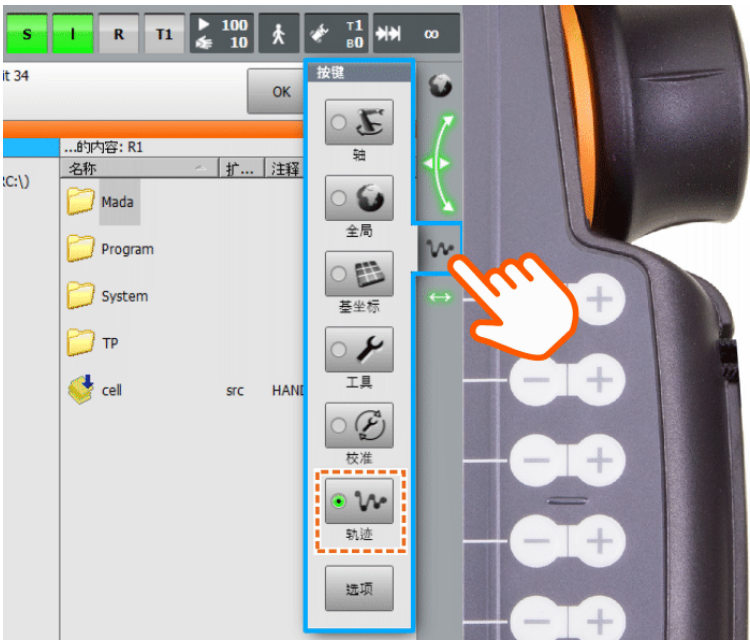


图 3-24：轨迹运行方式

2. 设定手动倍率。

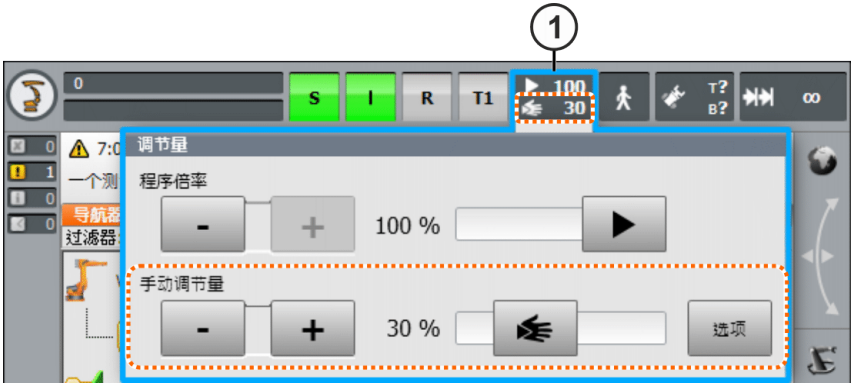


图 3-25：通过状态栏设定手动倍率

3. 按下一个使能开关并按住。



图 3-26: 使能开关

4. 除了最上面的移动键，还显示以下双箭头符号：



图 3-27: 轨迹手动运行

- 5. 对于该点动键，按下负号，使机器人沿记录的轨迹反向运行。
 - 6. 按下正号，使机器人沿记录的轨迹正向运行。
- 机器人可以交替地前进和回退。如果在一个或另一个方向上达到所记录轨迹的终点，则机器人控制器显示以下信息：*已到达记录的轨迹的末端*。

比较“启动反向”和“轨迹运行方式”

该表显示通过启动反向键和通过点动键反向运行之间的主要区别。

通过启动反向	通过点动键
可以反向运行程序运动。	可以反向运行几乎所有种类的运动。
轨迹曲线可能与正向轨迹不同。	轨迹曲线与正向轨迹时相同。
在反向运行时不执行轨迹逼近或摆动。	

通过启动反向	通过点动键
可以逐个反向运行原运动。	轨迹是连续的。可以在任意位置上停止，但是不能逐个地（一个运动接着一个运动）反向运行原来的运动。

在启动时从缓冲器中删除

如果用户已经通过选项 **轨迹** 反向运行机器人，然后启动另一个运动，则会删除一部分缓冲器。

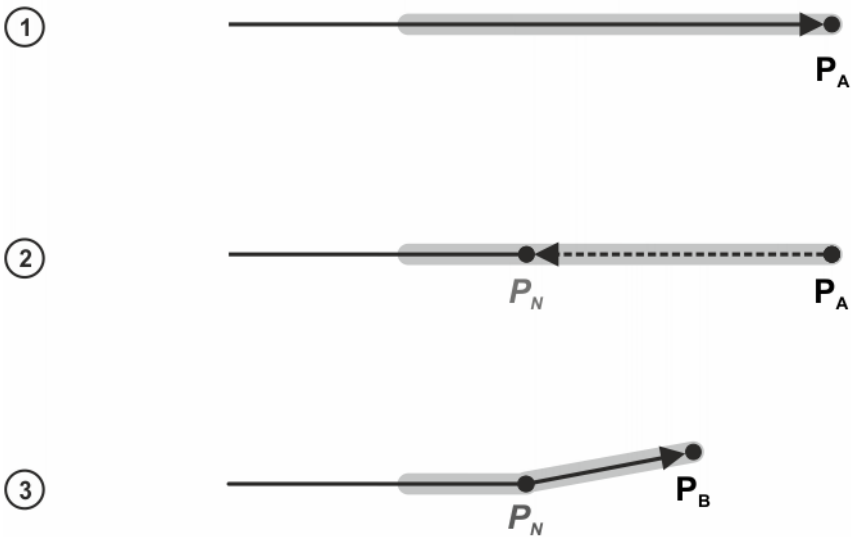


图 3-28：在启动时从缓冲器中删除

步骤	说明
1	初始状态：用户已经正向运行机器人。机器人位于一个点 P_A 上。在缓冲器中保存了前往此处的轨迹的最后一部分（灰色粗条柱）。
2	用户现在反向运行机器人（黑色的虚线箭头）。机器人现在位于缓冲器中的一个点 P_N 上。此时，在缓冲器中还保存了与第 1 步相同的轨迹。
3	用户在点 P_N 上开始另一个运动。指的是除了所记录轨迹上的继续运行之外的所有运动（反向还是正向无关紧要）。 机器人现在位于一个点 P_B 上。 - 从缓冲器中删除了从 P_N 至 P_A 的部分。 - 记录了从 P_N 至 P_B 的新运动。

3.8 机器人位置的显示方法

机器人位置的显示方式

当前的机器人位置可通过两种不同方式显示：

- 轴坐标：

```
$AXIS_ACT={A1..., A2..., A3..., A4..., A4..., A5..., A6...}
```

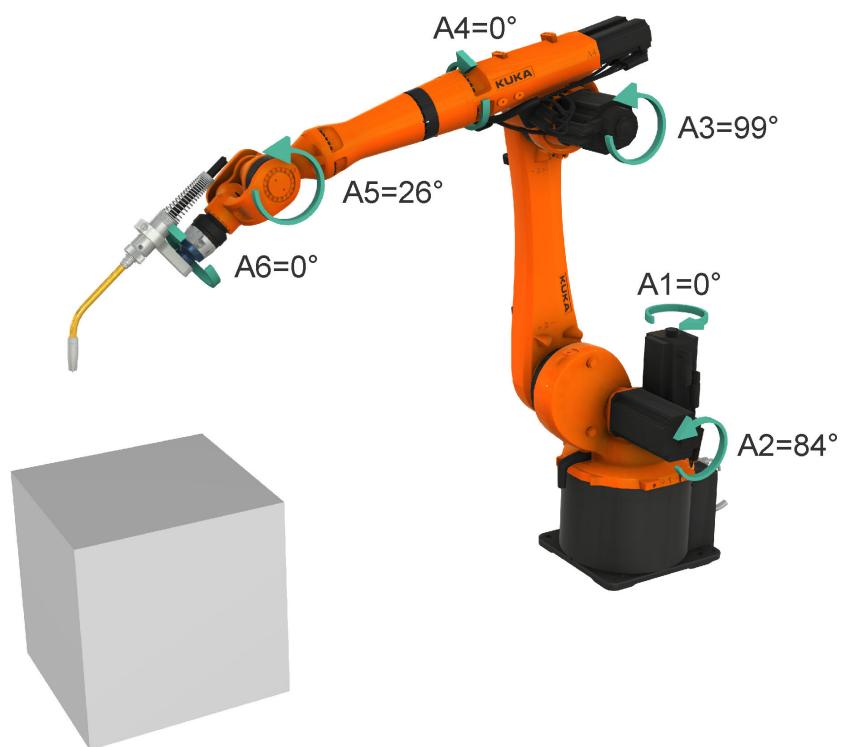



图 3-29：轴坐标中的机器人位置

- 显示每根轴的当前轴角：
- 该角等于与轴的机械零位之间的角度绝对值。
- 笛卡尔式：

```
$POS_ACT={X..., Y..., Z..., A..., B..., C..., S..., T..., E1...,...}
```

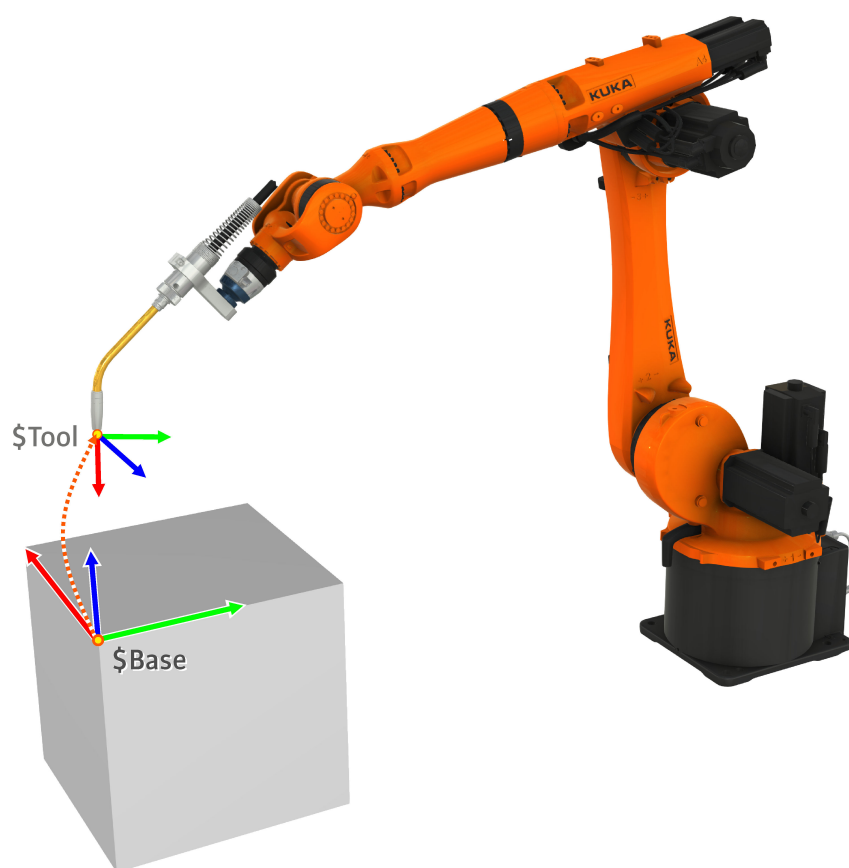


图 3-30：笛卡尔位置

- 显示当前 TCP（工具坐标系）相对于当前所选的基坐标系的当前位置。
- 没有选择工具坐标系时，法兰坐标系适用！
- 没有选择基坐标系时，世界坐标系适用！

不同基坐标系中的笛卡尔位置

观察下图时，我们会立即意识到，机器人的三个位置都相同。位置指示器在这三种情况下显示不同的值：

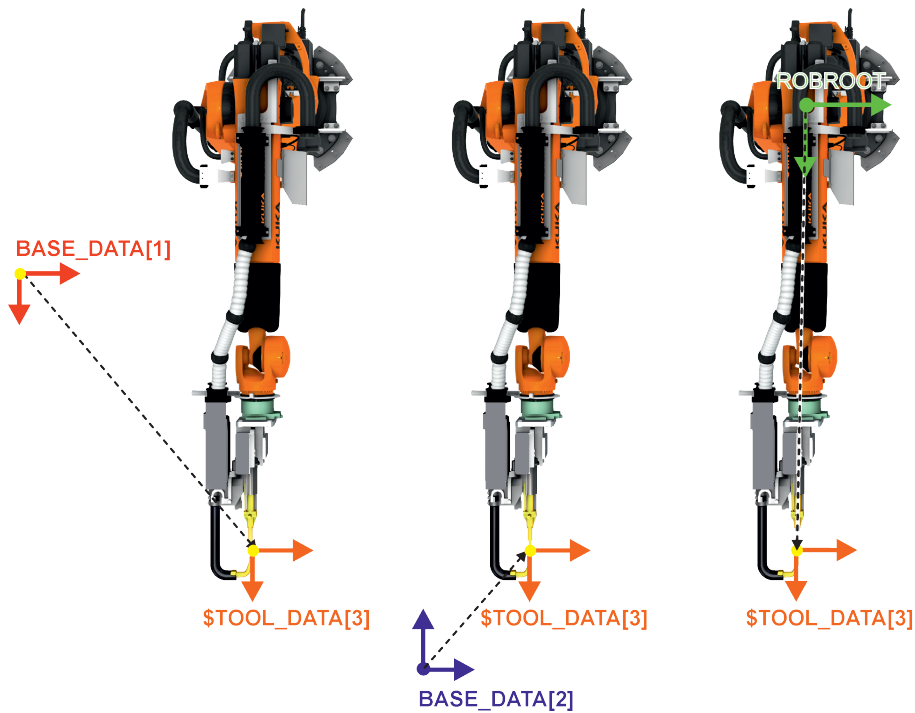


图 3-31：三个工具位置 - 一个机器人工位！

在相应的基坐标系中显示工具坐标系/TCP 的位置：

- 对于基坐标系 1
- 对于基坐标系 2
- 对于基坐标系 0：这相当于世界坐标系（通常也就是 \$NULLFRAME）！



仅当选择了正确的基坐标系和正确的工具时，笛卡尔坐标系中的实际位置指示器才显示所预期的值！

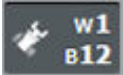


图 3-32：选择工具和基坐标

3.8.1 在 smartPAD 上查询当前机器人位置

说明

- 可以在两种类型的 smartPAD 上查询当前机器人位置。
 - 极坐标式位置
 - (>>> “以极坐标方式显示机器人位置” 页面 76)
 - 笛卡尔坐标式位置
 - (>>> “显示笛卡尔机器人位置” 页面 77)
- 机器人位置可以在每个用户组中和每个运行方式下显示。



也可以与一个程序同时或在手动运行模式下显示机器人位置。

操作步骤

以极坐标方式显示机器人位置

1. 菜单路径：机器人按键 > 显示 > 实际位置
2. 通过按键笛卡尔坐标式/极坐标式（7）可在视图之间切换。
3. 此外，可通过电机按键（6）在列（4）中显示电机角（3）。

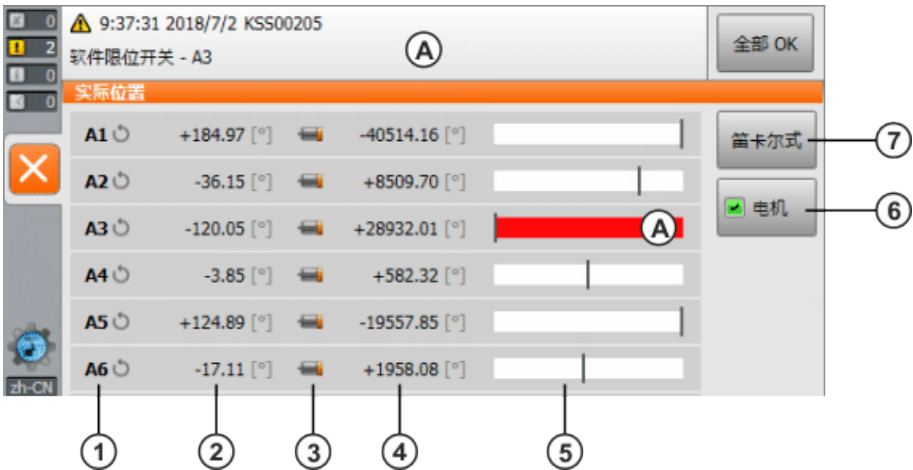


图 3-33： 机器人位置，极坐标的

序号	说明
1	轴名称和轴类型的符号显示 ：旋转式 ：无限旋转 ：线性
2	当前轴位置
3	“电机”符号 只在勾选了复选框电机时显示。
4	当前电机角度 只在勾选了复选框电机时显示。
5	对于旋转和线性轴：图形显示当前位置，基于允许的轴范围 - 条柱表示允许的范围，即软件限位开关之间的范围。左边缘相当于负限位开关，右边缘相当于正限位开关。条柱的中心相当于允许范围的中心。示例：负限位开关在 60° 上，正限位开关在 100° 上。因此，条柱的中心相当于 20° 。 - 垂直线条表示轴在该范围中的位置。 - 如果轴达到了限位开关，则条柱变为红色并且在信息窗（A）中输出一条信息。 对于无限旋转轴，显示有偏差。

显示笛卡尔机器人位置

- 菜单路径： 机器人按键 > 显示 > 实际位置
- 通过按键笛卡尔坐标式/极坐标式（7）可在视图之间切换。

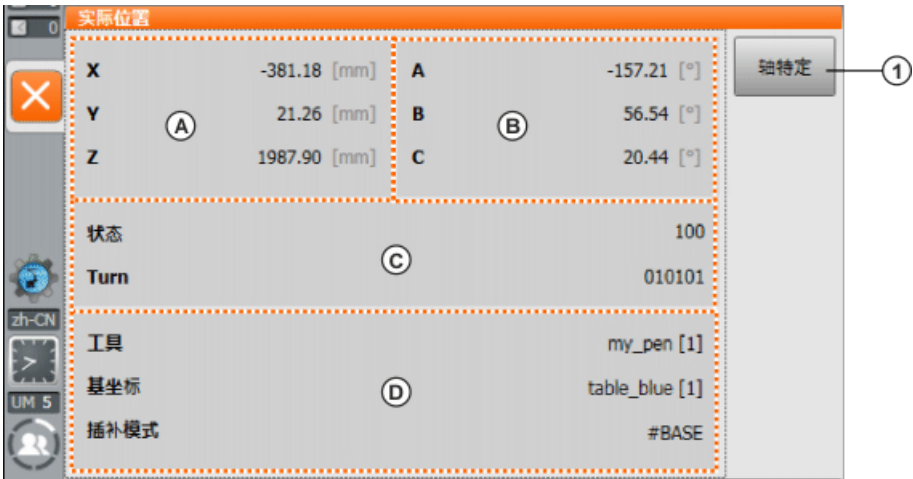


图 3-34：机器人位置，笛卡尔式

序号	显示栏	说明
A	X、Y、Z	当前位置
B	A、B、C	当前姿态
C	状态	二进制表示的状态，3 位
	Turn	二进制表示的转角，6 位
D	工具	- 有效工具或有效基坐标的名称和编号或者 \$NULLFRAME[0] - 如果工具/基坐标都无效，例如在控制系统启动之后，则显示“?[-1]”。
	基坐标	
	插补模式	#BASE：该工具已安装在连接法兰处。 #TCP：该工具为一个固定工具。 (>>> “用固定工具时更改的运动过程” 页面 327)

4 机器人的投入运行

4.1 学习单元：机器人投入运行

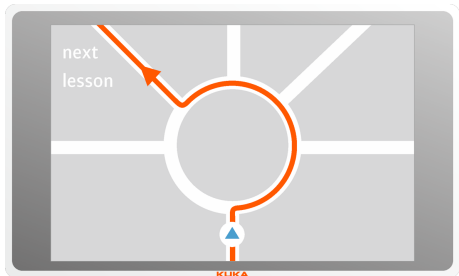


图 4-1：导航器

本学习单元讲授以下内容：

- 控制系统上的用户组
- 零点标定的原理
- 工具和基坐标管理
- 机器人上的负载
- 工具测量
- 基坐标测量
- 显示内存使用情况
- 关闭控制系统

4.2 控制系统上的用户组

说明



图 4-2：用户组

- 通过用户组可以在机器人控制器上有针对性地为不同的人群分配功能和权限。
- 除了默认用户组之外，所有其他用户组都有密码保护。
所有组的默认密码为 “kuka”。
所有组的默认密码为 “kuka”。
- 为了防止滥用，要为每个组单独更改密码。



在这一方面，管理员、安全投入运行人员和安全维护人员用户组极其敏感。

- 可通过权限管理为每个用户组分配其他权限。权限管理仅供管理员用户组使用。

用户组

- **操作人员**
很有限的权限：操作人员不允许执行会永久更改系统的功能。
- **应用人员**
应用人员允许执行机器人正常运行所需的功能。
示例：借助行指令编程简单的运动程序。
- **专家**
专家允许执行专业技术知识所需的功能。
示例：专家程序的编程。
- **安全维护人员**
该用户组可以借助激活代码激活控制系统上准备好的安全接口（通过安全投入运行人员）或人员安全的培训单元配置（例如 SafeOperation）。
安全维护人员拥有更多权限。
- **安全调试人员**
该用户组有权更改控制系统上的安全接口或配置人员安全的培训单元配置（例如 SafeOperation）。
- **管理员**
管理员允许执行所有功能（包括安全技术）。
该用户组拥有访问权限管理的权限。

4.2.1 切换用户组

当前用户组的显示

- 在 smartHMI 上，符号 **用户组** 通过其白色弧段的数量显示当前选择了哪个用户组。
- 示例：3 个白色弧段 = **专家**



图 4-3：用户组：专家

操作步骤

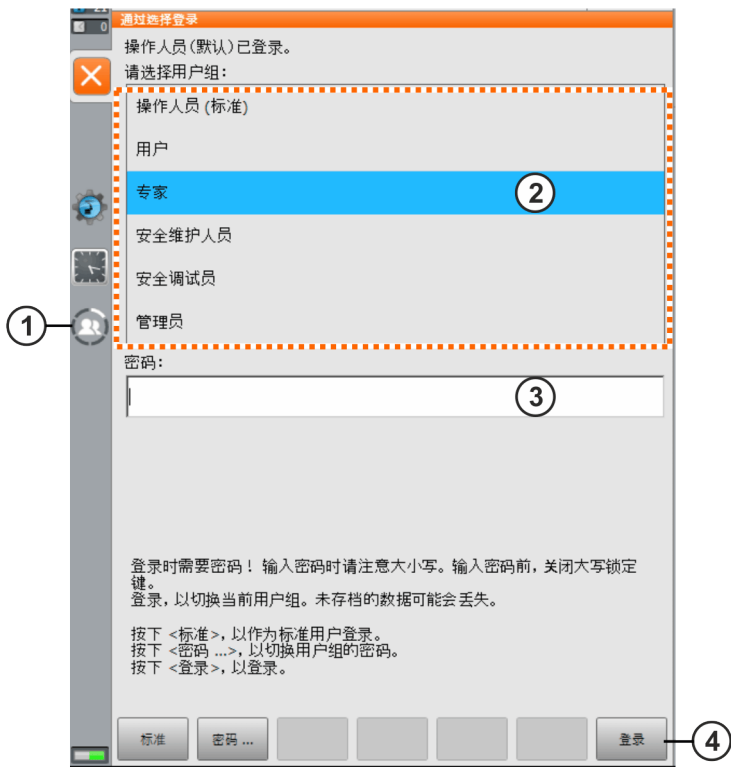


图 4-4: 切换用户组

1. 在 smartHMI 上触摸符号 **用户组** (1)。
可选: 选择**机器人按钮** > **配置** > **用户组**。
用户组选择窗口打开。
2. 切换用户组:
 - 如要切换到默认用户组, 按下**标准**。
如果当前已为默认用户组, 则不显示**标准**。
 - 为了切换到另一个用户组, 选中所需的用户组 (2)。
然后输入密码 (3) 并用 **登录** (4) 确认。

4.2.2 更改密码

操作步骤

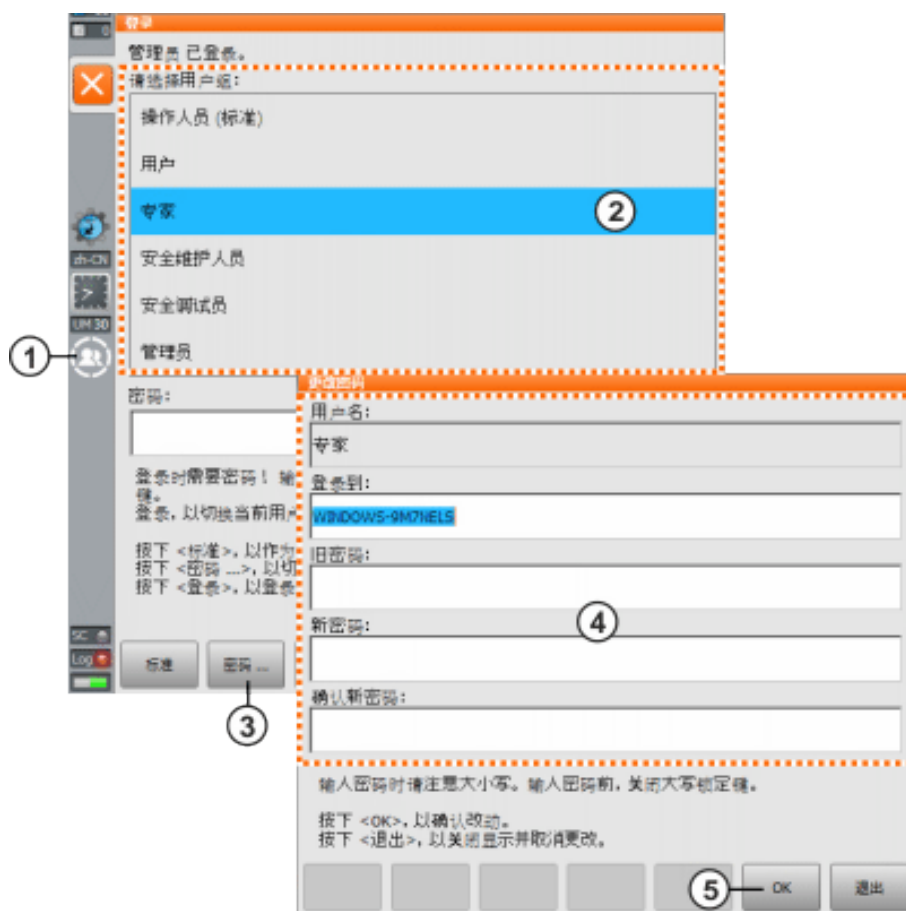


图 4-5：更改密码

1. 在 smartHMI 上触摸符号 **用户组** (1)。
或者：在主菜单中选择**配置** > **用户组**。
用户组选择窗口打开。
2. 选中要更改其密码的用户组 (2)。
3. 在按钮栏的下部按下 **密码 ...** (3)。
4. 输入旧密码。输入新密码两次。(4)
出于安全考虑，输入内容均会加密显示。密码区分大小写。
5. 按下 **OK** (5)。新密码立刻生效。

4.2.3 在控制系统上管理权限

说明



图 4-6

- 只有**管理员**用户组可以使用控制系统上的权限管理菜单项。

菜单路径: 机器人按键 > 投入运行 > 权限管理



所有用户组都有读取权限。

- 通过**权限管理**确定，
 - 哪些**运行方式**可供各个用户组使用。



图 4-7: 权限管理，运行方式

- 哪些**功能组**可供各个用户组使用。

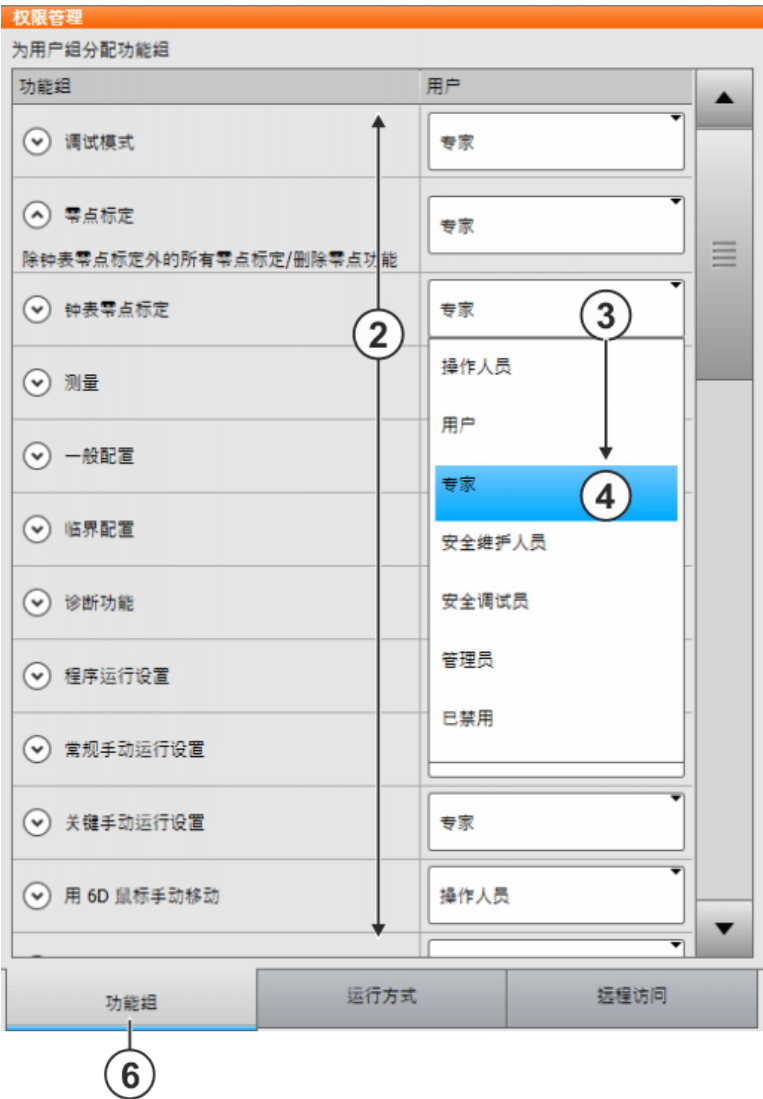


图 4-8：功能组

- 1 功能组选项卡
- 2 可分配给特定用户组的功能
- 3 显示当前用户组
- 4 设定新用户组

4.3 零点标定的原理

为什么要标定零点？



图 4-9

零点标定时，会给每个机器人轴分派一个基准值。这样机器人控制器可识别到轴位于何处。

仅在工业机器人得到充分和正确标定零点时，它的使用效果才会最好。只有这样机器人才能达到它最高的点精度和轨迹精度。可以执行程序。仅可轴特定地移动未标定零点的机器人。

完整的零点标定过程包括为每一个轴标定零点。通过技术辅助工具（EMD – *Electronic Mastering Device*，即电子测量仪）可为任何一根在**机械零点位置**的轴指定一个机器人特定的基准值（例如：0°、-90°、110° 等）。因为这样就可以使轴的机械位置和电气位置保持一致，所以每一根轴都有一个唯一的角度值。所有机器人的零点标定位置都是相似的，但不尽相同。精确位置在同一机器人型号的不同机器人之间也会有所不同。



图 4-10：零点标定位置

机械参考点位置角度值示例：

轴	Cybertech KR16 R1610	QUANTEC KR120 R2700-2
A1	27°	-25°
A2	-90°	-100°
A3	90°	100°
A4	0°	0°
A5	0°	0°
A6	0°	0°

何时标定零点？

原则上，机器人必须时刻处于已标定零点的状态。在以下情况下必须进行零点标定：

- 在将机器人投入运行时



图 4-11: 机器人投入运行

- 在对参与位置值检测的组件（例如带旋转变压器或 RDC 的电机）采取了维护措施之后。

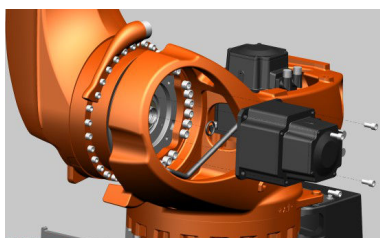


图 4-12: 更换电机

- 当未用控制系统（例如借助于释放装置）移动了机器人轴时。



图 4-13: 自由旋转装置

- 在机械维修/出现问题之后，根据不同的要求进行零点标定或负载零点标定，并应用测得的偏差。
 - 更换齿轮箱后

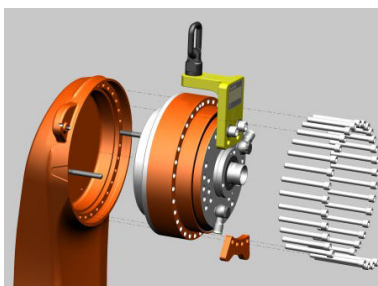


图 4-14: 更换，齿轮箱

- 以高于 250 mm/s 的速度撞到机械限位上之后



图 4-15：撞到终端止挡上

— 在碰撞后



图 4-16：碰撞



在进行维护前一般应检查当前的零点标定。

关于零点标定的安全提示

如果机器人轴未经零点标定，则会严重限制机器人的功能：

- **无法编程运行**
无法接近编程设定的点。
- **无法进行笛卡尔式手动运行**
不能在坐标系中移动。
- **软件限位开关已停用**
因此可能会撞到机械终端止挡上。

4.3.1 零点标定工具

说明

对于机器人的零点标定，有两个零点标定工具可用：

- 千分表
(>>> “千分表” 页面 88)
- 零点标定组件
(>>> “SEMD/MEMD 校准套件” 页面 88)

千分表

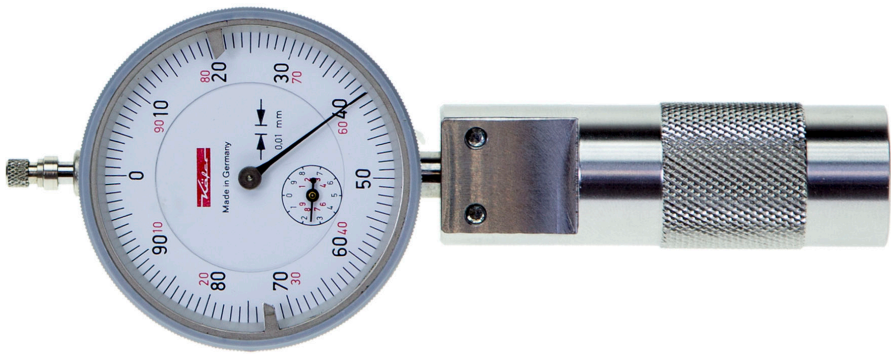


图 4-17：千分表

- 可借助千分表可“标准”标定机器人的零点。
- 精度很大程度上取决于零点标定工具（例如：视差误差）。
- 这与 SEMD/MEMD—零点校准组件有关。
(>>> “SEMD/MEMD 校准套件” 页面 88)

提示

无法用千分表进行负载零点标定。

SEMD/MEMD 校准套件



图 4-18：SEMD/MEMD 校准套件

- 1 电缆
 - 2 通用型校准箱
 - 3 MEMD 传感器，用于小型校准套筒
 - 4 SEMD 传感器，用于大型校准套筒
- SEMD/MEMD 校准套件可代替下列校准套件：
 - EMT – KR C1 和 KR C2 的电子测量工具

- 为此需要 **KR C2 技术选项**（适配器）。
 - **EMD** – KR C4 技术的 **电子控制装置**
- 借助 SEMD/EMD 校准套件可对所有配备**大型**和**小型**校准套筒的机器人执行零点标定。
 - **MEMD** – **微电子测量仪**
用于小型校准套筒
 - **SEMD** – **标准电子测量仪**
用于大型校准套筒
- 该校准组件也可独立作为 SEMD 或 MEMD 校准套件供货。

可用的校准组件

校准组件	货号	说明
SEMD/ MEMD	<u>00-228-936</u>	用于所有测量套筒的校准设备 (M8/M20 细螺纹) 可用于所有机器人型号
SEMD	<u>00-228-934</u>	用于 M20 细螺纹测量套筒的校准设备 可用于例如 QUANTEC 系列
MEMD	<u>00-208-642</u>	用于 M8 细螺纹测量套筒的校准设备 可用于例如 AGILUS
KR C2 选项	<u>00-228-327</u>	用于将 SEMD 用作 KR C2 系统的校准设备 (KTL) 的适配器电缆。

使用 MEMD/SEMD



图 4-19: SEMD 连接示例

4.3.2 确定机械零位

零点标定过程

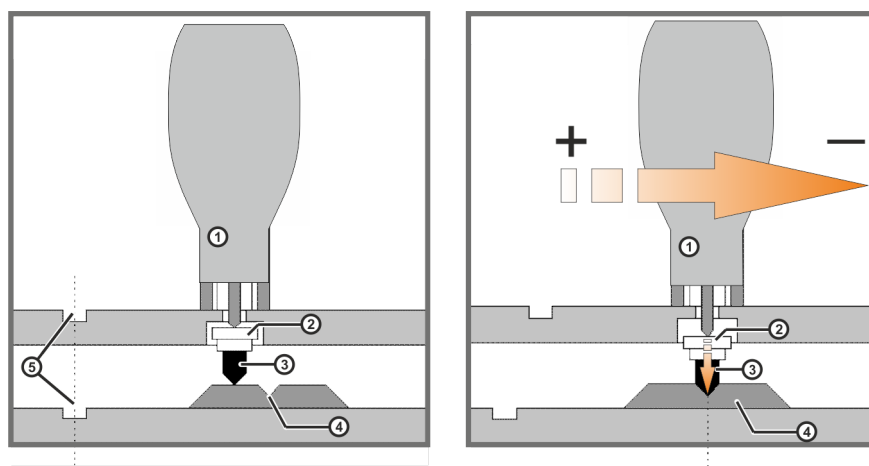


图 4-20: EMD 校准流程

- | | |
|---------------|-----------|
| 1 EMD (电子测量仪) | 4 测量槽 |
| 2 测量口 | 5 预零点标定标记 |
| 3 探针 | |

机械零点的计算

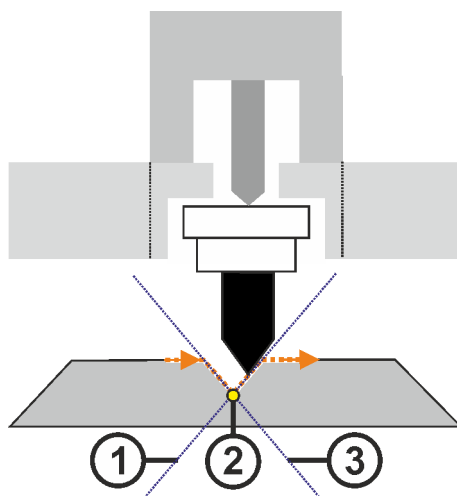


图 4-21

进行零点标定，确定轴的机械零点 (2)。此时轴会移动，直到探针驶过零点标定槽口一次。然后确定槽口的精确位置。计算下沿 (1) 和上沿 (3) 的交点，为最低点 (2)。最低点 (2) 为轴的机械零位。将该轴调整为在设备数据中保存的零点标定值 (度数数据)。因此，每根轴都配有一个测量筒、一个测量槽和一个预零点标定标记。



对于多种机器人类型如 Agilus 1 和 KR CYBERTECH nano，轴 6 没有测量筒。对于这些机器人类型，借助所涉轴上的标记进行机械归零。

4.3.3 标定机器人零点

机器人的零点标定方式

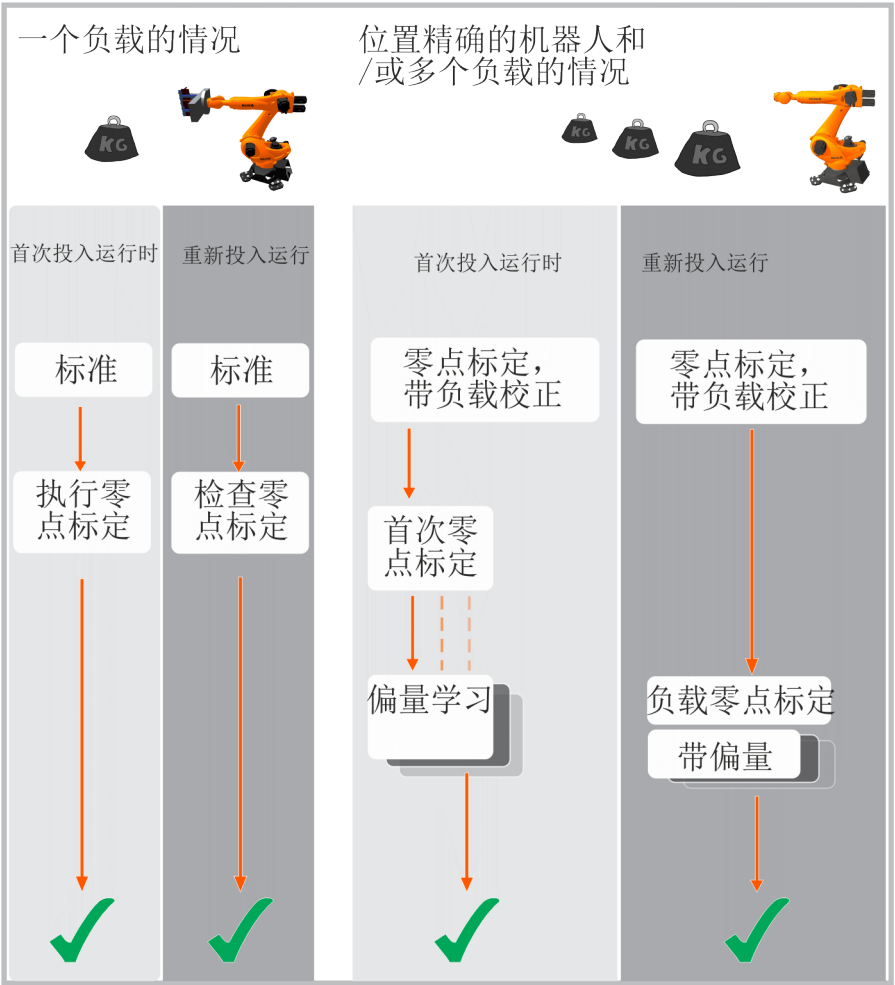


图 4-22: 零点标定途径

• 标准零点标定

图片栏：负载实例

机器人操作重量恒定且固定安装的工具（例如焊嘴、点焊钳、空抓爪）时，将使用该零点标定方式

对可能的重新投入运行的影响：

- 如果在创建程序之后用已更改的工具或负载标定机器人的零点，则这对所示教点的精度有直接影响。
- 更改的负载将导致更大/更小的机器人“翘曲”和要保存的已更改旋转变压器值。
- 保存的旋转变压器值本身是确定机器人位置的计算基础（“开始位置”）。

• 零点标定，带负载校正

图片栏：位置精确的机器人和/或多个负载情况

以下情况使用该零点标定方式：

- 在没有拆卸工具和附加负载的情况下要恢复首次零点标定（不带负载）。*)
- 机器人在其应用领域中用工具更换系统例如抓手和点焊钳进行工作。



*) 定位精确的机器人模型以不带负载的首次零点标定为基础。因此，在每次零点标定时恢复该状态非常重要。为了简化该过程和避免工具和附加负载的拆卸，有通过负载修正进行零点标定的行程可用。

对可能的重新投入运行的影响

- 在投入运行期间始终对机器人进行首次零点标定（不带工具/负载），然后为每个工具示教偏差量。
- 按照工具保存在此确定的首次零点标定偏移量。
- 根据故障情况，可以用当前安装的工具/负载标定机器人的零点。
- 由于工具特定的偏移量，机器人可以反算出首次零点标定。

4.3.3.1 首次/标准零点标定

操作步骤

首次零点标定



图 4-23：首次零点标定

提示
只有当机器人没有负载时才可以执行首次零点标定。不得安装工具和附加负载。

1. 将机器人移到预标定位置。

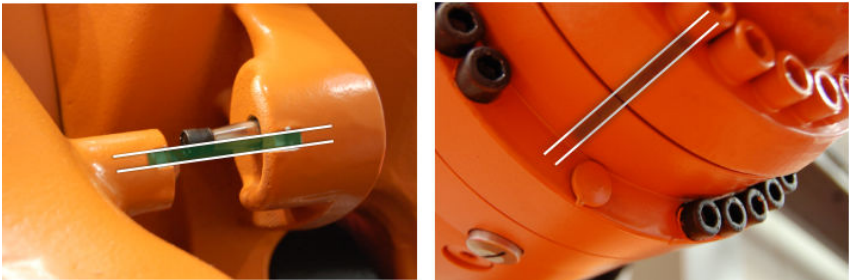


图 4-24：预零点标定位置示例

- 设置缓慢的手动倍率。
- 借助 $\pm$ 按键缓慢地将待零点标定的轴移至机械对准标记上。

- 2. 菜单路径：机器人按键 > 投入运行 > 零点标定 > EMD > 带负载修正 > 首次零点标定
- 3. 首次零点标定窗口

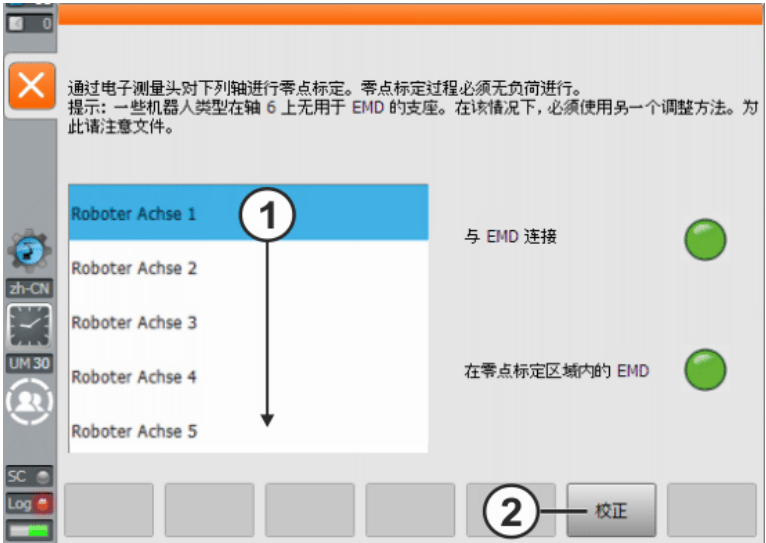


图 4-25：首次/标准零点标定

- 在菜单（1）中点击待零点标定的轴。
已激活的轴蓝色突出显示。
- 4. 将 MEMD/ SEMD 螺接并连接到测量口上。

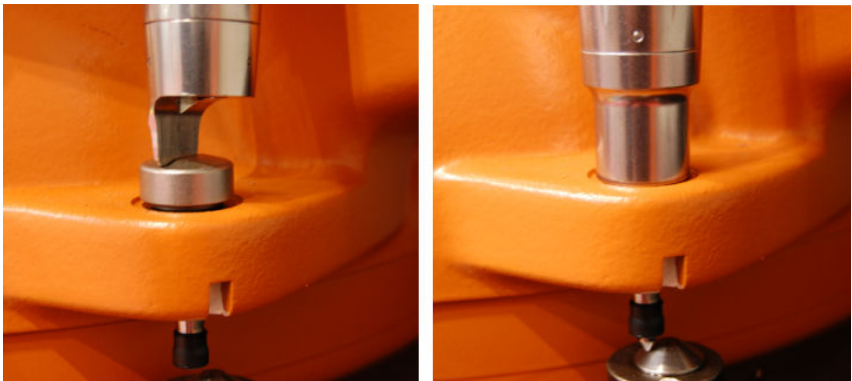


图 4-26：EMD，已拧到测量筒上

- 从窗口中选定的物理轴上取下测量口的防护盖。
为此可以使用零点标定组件中随附的螺丝刀。
- 5. 连接 MEMD/SEMD 导线

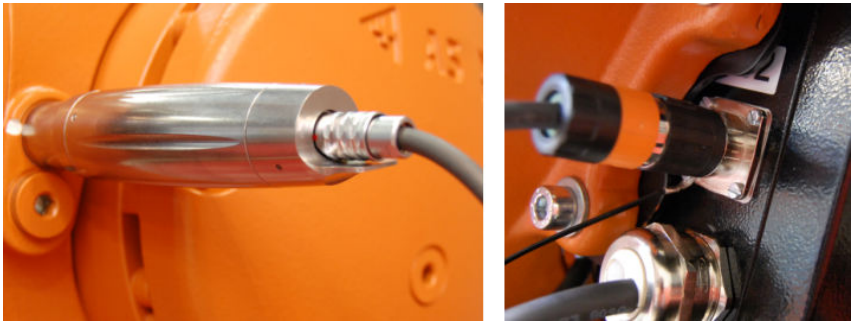


图 4-27：EMD 电缆，连接

- 将 MEMD/SEMD 拧到测量口上。



- 在 EMD 上和底座接线盒的接口 X32 上连接测量导线。

小心

始终将 EMD 不带测量导线拧到测量口上。然后方可将测量导线接到 EMD 上。否则测量导线会被损坏。
同样在拆除 EMD 时也必须先拆下 EMD 的测量导线。然后才将 EMD 从测量口上拆下。
在标定之后，将测量导线从接口 X32 上取下。否则会出现干扰信号或导致损坏。

6. 进行零点标定运行，示教偏量。

- 用零点标定按键 (2) 启动零点标定运行。
- 在信息窗口中显示信息 **需要启动键**。



图 4-28: 信息窗口

7. 按下使能开关和启动键。



图 4-29: smartPAD 上启动键的位置

- 机器人离开零点标定槽口并确定零点标定位置。
在整个测量运行期间要按住这些按键。
 - 机器人自动停止运行。
- 在选择窗口中刚刚经零点标定的轴被隐藏。
 - 将测量导线从 EMD 上取下。然后从测量口上取下 EMD，并将防护盖重新装好。
 - 为所有待零点标定的轴重复这些步骤。
 - 关闭窗口。
 - 将测量导线从接口 X32 上取下。

标准零点标定



图 4-30：标准零点标定的机器人

提示

与首次零点标定不同，用已安装的工具测量标准零点标定。

- 菜单路径：机器人按键 > 投入运行 > 零点标定 > EMD > 标准 > 设定零点标定
- 所有其他步骤与首次零点标定相同。
(>>> “首次零点标定” 页面 92)

提示

- 如果已经对机器人进行了首次零点标定和示教了偏移量，则不允许用已安装的工具进行标准零点标定。
- 在控制系统内的相同位置上保存首次和标准零点标定。
- 无法再反算为各种工具！
- 无法再精确地驶向编程设定的点。

4.3.3.2 零点标定，带负载校正

钓鱼人在日常生活中遇到的情况

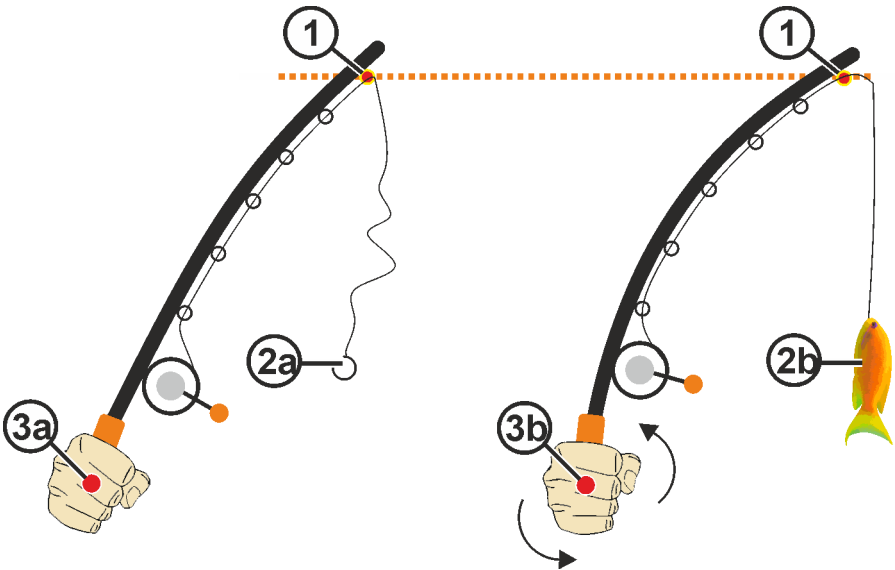


图 4-31：负载对旋转点的影响

钓鱼人想钓到一条鱼。为此，他将钓竿放到水面上方，努力找到一个与水面成规定距离的固定点 (1)。为此他会“转动”自己的手腕，直至到达这个位置。此时钓竿未负载 (2a)，“还没有鱼上钩”。如果有鱼上钩 (2b)，则钓竿会受到负载并导致翘曲/弯曲，因为钓竿具有弹性。为了保持起始点 (1)，必须“继续转动”手腕 (3b)。

为何要学习偏量?



图 4-32：齿轮箱间隙和弹力

固定在法兰上的工具的重量在较大的杠杆作用影响下，会给机器人带来静态载荷。由于部件和齿轮箱上材料固有的弹性，承载的机器人与未承载的机器人相比，其位置会有所区别。这些相当于几个增量的区别将影响到机器人的精确度。



增量是最小的数字步幅，以检测机器人电机上的旋转。增量类似于分度。



图 4-33：偏量学习

“偏量学习”即带负载校正。与首次零点标定（无负载）的差值被储存。

- 如果机器人以各种不同负载工作，则要对每个负载都执行“偏量学习”功能，以便更容易恢复零点标定。
- 对于抓取重型工件、需要简化恢复首次零点标定的抓爪，必须对抓爪分别在不带工件和带工件时进行“偏量学习”。



在使用现有设备时，可使用的零点标定方式（标准或带负载校正）必须与调试时所使用的零点标定方式相符。如果使用错误的零点标定方式，将导致机器人的零点标定错误。

偏量学习操作步骤



“偏量学习”即带负载校正。保存与首次零点标定之间的差值。

1. 已在不带工具和负载的机器人上进行了首次零点标定。

菜单路径：机器人按键 > 投入运行 > 零点标定 > EMD > 带负载修正 > 首次零点标定



图 4-34：首次零点标定

2. 在机器人法兰上拧上零点标定所需的工具。

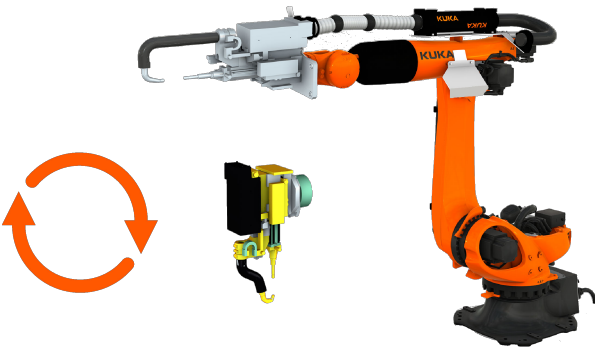


图 4-35：带工具的零点标定

3. 将机器人移到预标定位置。

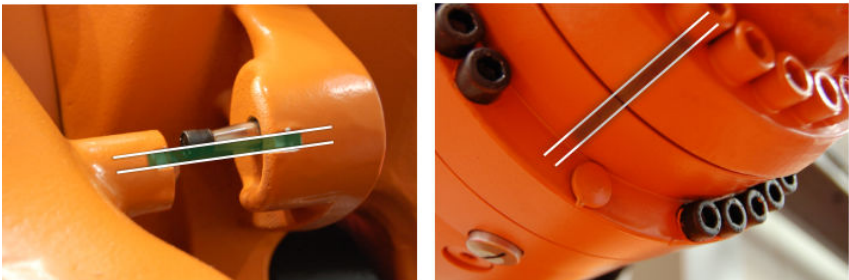


图 4-36：预零点标定位置示例

- 设置缓慢的手动倍率。
 - 借助 $\pm$ 按键缓慢地将所有待零点标定的轴移到机械对准标记上。
4. 菜单路径：机器人按键 > 投入运行 > 零点标定 > EMD > 带负载修正 > 示教偏移量
5. 输入工具编号。



图 4-37：负载零点标定的工具选择

- 在输入栏中用数字输入工具编号。
- 用按键工具 OK 确认。

6. 偏量学习零点标定窗口



图 4-38：偏量学习

- 在菜单 (1) 中点击待零点标定的轴。
已激活的轴背景色被置为蓝色。

7. 将 MEMD/ SEMD 螺接并连接到测量口上。

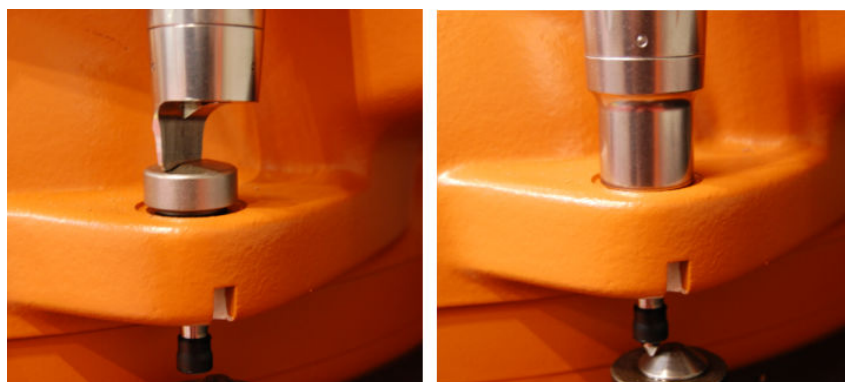


图 4-39：EMD，已拧到测量筒上

- 从窗口中选定的物理轴上取下测量口的防护盖。
为此可以使用零点标定组件中随附的螺丝刀。

8. 连接 MEMD/SEMD 导线

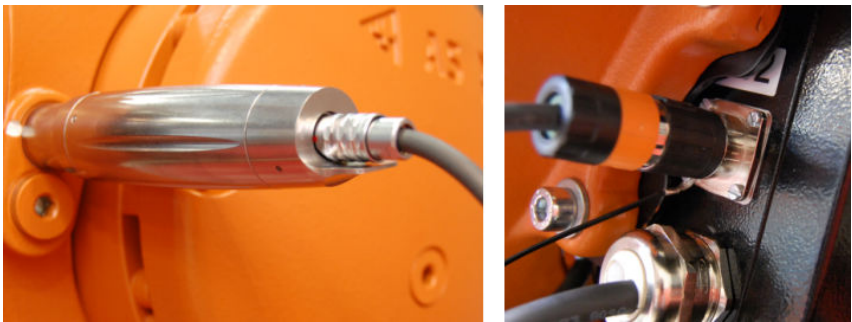


图 4-40：EMD 电缆，连接

- 将 MEMD/SEMD 拧到测量口上。
- 在 EMD 上或底座接线盒的接口 X32 上连接测量导线。



小心

始终将 EMD 不带测量导线拧到测量口上。然后方可将测量导线接到 EMD 上。否则测量导线会被损坏。
同样在拆除 EMD 时也必须先拆下 EMD 的测量导线。然后才将 EMD 从测量口上拆下。
在标定之后，将测量导线从接口 X32 上取下。否则会出现干扰信号或导致损坏。

9. 进行零点标定运行。

- 用示教按键 (2) 启动零点标定运行。
- 在信息窗口中显示信息 *需要启动键*。



图 4-41：信息窗口

10. 按下使能开关和启动键。



图 4-42：smartPAD 上启动键的位置

- 机器人自动离开零点标定槽口并确定零点标定位置。
在整个测量运行期间要按住这些按键。
 - 机器人自动停止运行。
11. 按下使能开关和启动键。



图 4-43：电机角度/轴位置差

- 一个窗口打开。
该轴上与首次零点标定的偏差以电机角度和轴角度形式显示 (1)。
- 用 OK 按键保存值并且可以标定下一个轴的零点。
- 在选择窗口中刚刚经零点标定的轴被隐藏。
(>>> 图 4-38)

- 12. 将测量导线从 EMD 上取下。然后从测量口上取下 EMD，并将防护盖重新装好。
- 13. 为所有待零点标定的轴重复这些步骤。
- 14. 将测量导线从接口 X32 上取下。
- 15. 用关闭来关闭窗口。

故障后的负载零点标定

- 如果在故障/碰撞之后更换电机或齿轮箱，则必须重新标定机器人的零点。
- 原则上必须检查机器人处于哪个状态下。
 - 机器人上无工具。
菜单路径：机器人按键 > 零点标定 > EMD > 带负载修正 > 负载零点标定 > 首次零点标定
测量过程相当于首次投入运行“首次零点标定”
(>>> 4.3.3.1 “首次/标准零点标定” 页面 92)
 - 机器人上有工具
菜单路径：机器人按键 > 零点标定 > EMD > 带负载修正 > 负载零点标定 > 带偏移量
测量过程相当于首次投入运行“带负载修正的零点标定”
(>>> 4.3.3.2 “零点标定，带负载校正” 页面 95)

提示

仅当机器人未进行机械更改时，才能使用不带偏移量的负载零点标定。在碰撞、电机更换、齿轮箱更换等时不允许进行此操作

4.3.3.3 练习：零点标定，带偏移量的负载零点标定

练习内容

打开课程练习本并完成以下练习：

- 零点标定，带偏移量的负载零点标定

4.3.4 小型机器人 AGILUS 上的零点标定

说明

KR AGILUS 1 上的零点标定位置

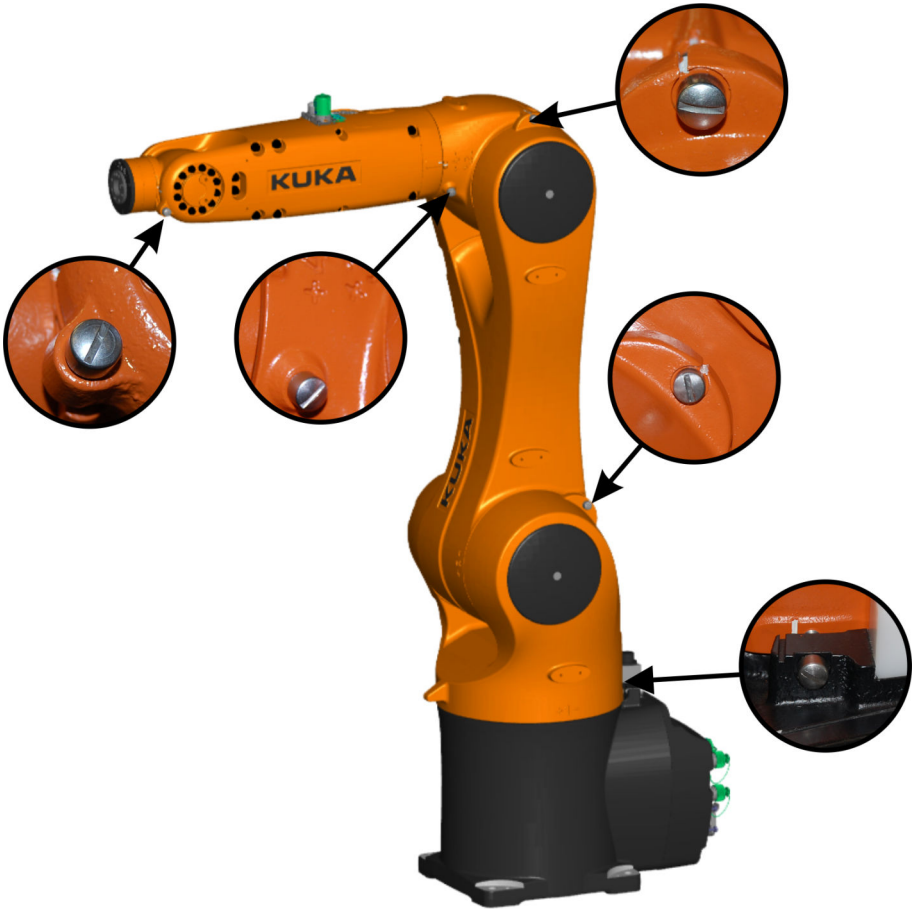


图 4-44：KR AGILUS 的零点标定位置

机械零点位置的角度值 (= 基准值)

轴	KR AGILUS 1
A1	0°
A2	-90°
A3	+90°
A4	0°
A5	0°
A6	0°

操作步骤

- 使用用于小型校准套筒的 MEMD 传感器为轴 1 - 5 进行零点标定。
- 如果机器人的 A6 上没有常规的零点标定标记，而采用标记线，则在没有 MEMD 的情况下对 A6 进行零点标定。
- 在零点标定之前，必须将 A6 移至零点标定位置。（所指的是在总零点标定过程之前，而不是直接在 A6 自身的零点标定前）。为此 A6 的金属上刻有很精细的线条。
- 为了将 A6 移至零点标定位置上，这些线条要精确地相互对齐。



在驶向零点标定位置时，须从前方正对着朝固定的线条看，这一点尤其重要。如果从侧面朝固定的线条看，则可能无法精确地将运动的线条对齐。后果是没有正确地标定零点。

- 菜单路径：选择**机器人按钮** > **投入运行** > **调整** > **参考**。
选项窗口**基准零点标定**打开。轴 A6 即被显示出来，并且被选中。
- 按下**校正**。轴 A6 即被标定零点并从该选项窗口中消失。
- 关闭窗口。

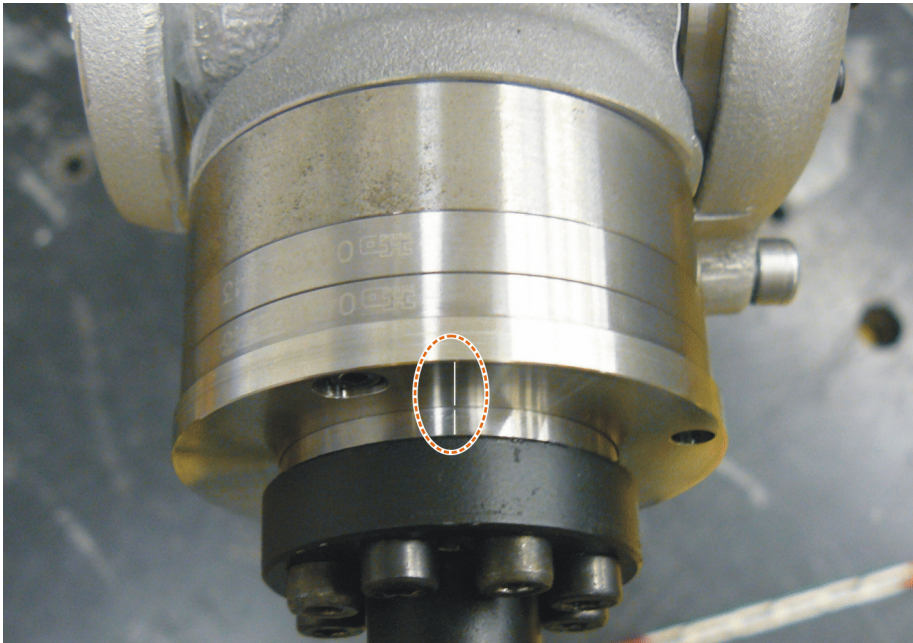


图 4-45：轴 A6 的零点标定位置 - 正面俯视图

KR AGILUS 2 上的零点标定位置



图 4-46：KR Agilus 2，零点标定套筒



在后续 KR Agilus 2 中有一个零点标定套筒进行轴 6 的零点标定。

4.3.5 重要的零点标定文件

零点标定 LOG 文件

- 如果对机器人进行零点标定，则将特定的零点标定数据记录到一个 LOG 文件中。
- 测定的偏移量以度为单位保存在 `C:\KRC\ROBOTER\LOG\Mastery.log` 中。

- 以下特定的零点标定数据保存在文件 **Mastery.log** 中：
 - 时间戳记（日期，时间）
 - 轴
 - 机器人的序列号
 - 零点标定值 (*FirstEncoderValue*)
 - 工具编号
 - 用度表示电机上的偏量值 (*Encoder Difference*)
- **Mastery.log 举例：**

```
Date: 01.09.11 Time: 13:41:07 Axis 1 Serialno.: 864585 First Mastering (First
Date: 01.09.11 Time: 13:42:07 Axis 2 Serialno.: 864585
First Mastering (FirstEncoderValue: 0.644334)
Date: 01.09.11 Time: 13:42:56 Axis 3 Serialno.: 864585
First Mastering (FirstEncoderValue: 0.745757)
Date: 01.09.11 Time: 13:43:29 Axis 4 Serialno.: 864585
First Mastering (FirstEncoderValue: 1.450234)
Date: 01.09.11 Time: 13:44:03 Axis 5 Serialno.: 864585
First Mastering (FirstEncoderValue: 0.686983)
Date: 01.09.11 Time: 13:44:30 Axis 6 Serialno.: 864585
First Mastering (FirstEncoderValue: 0.901439)
...
Date: 01.09.11 Time: 14:07:10
Axis 1 Serialno.: 864585
Tool Teaching for Tool No 1 (Encoder Difference: -0.001209)
Date: 01.09.11 Time: 14:08:44
Axis 2 Serialno.: 864585
Tool Teaching for Tool No 1 (Encoder Difference: 0.005954)
...
```

零点标定文件 (*.cal)

- **[序列号].cal** 保存在 RDC 的 EDS 上并且含有机器人的零点标定值。
 - 该文件通常无法查看，但是可以复制或替换。
- 为此，需要特殊的知识和更高的用户配置文件。

下列情况下，在 **C:\KRC\Roboter\RDC\[Ser. Nr.].cal** 还将保存校准位置

- 在 HMI 的菜单 **投入运行 > 机器人数据** 中按下 **“保存 RDC 数据”** 按钮。
- 控制系统重启之后 RDC 文件夹重新被删除。

```
[AbsolutMotorValues]
Axis1=0.463675
Axis2=16.257612
Axis3=36.209384
Axis4=66.48317
Axis5=20.9606263
Axis6=68.8641154

[MotorDifference for Tool 1]

[MotorDifference for Tool 2]

...
[MotorDifference for Tool 14]
Axis1=-0.157869
Axis2=0.553673
Axis3=-1.059499
Axis4=-0.169838
```

```
Axis5=-0.877667571
Axis6=-0.32594979

[MotorDifference for Tool 15]

...

[MotorDifference for Tool 128]

[CalibrationDifference]
Axis1=0.00116852
Axis2=-0.00313334352
Axis3=-0.0100344565
Axis4=0.0115672826
Axis5=-0.0113223559
Axis6=0.02928708
```

4.4 工具和基座管理

说明

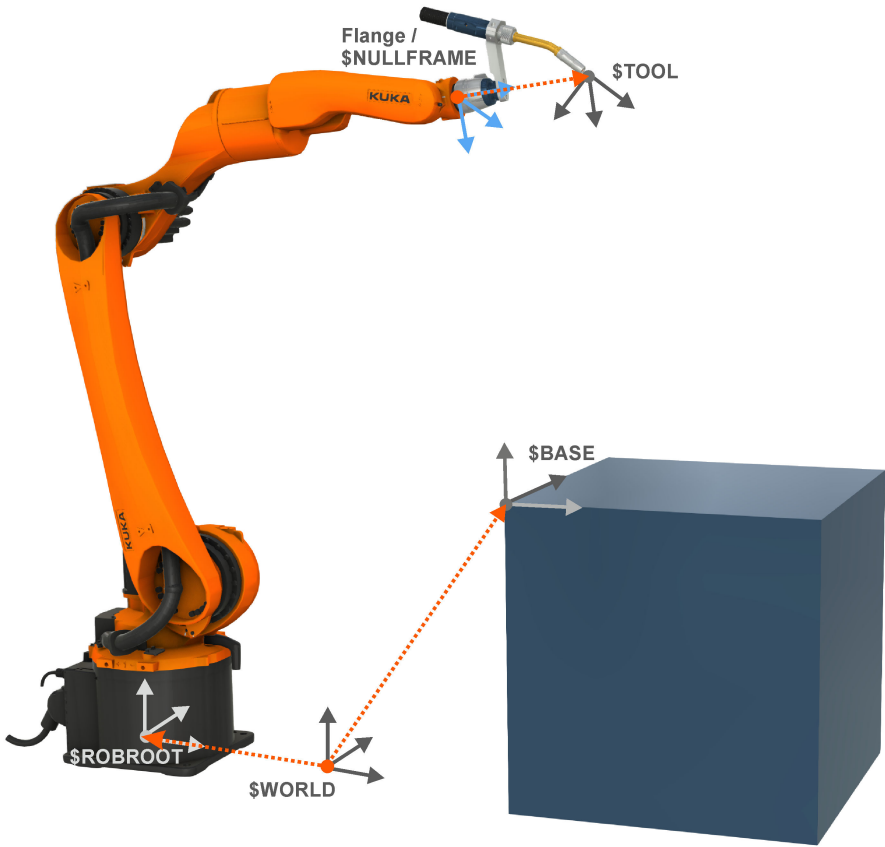


图 4-47: 坐标系统一览

- 通过工具/基座坐标管理插件，可以使用一个工具创建、编辑和删除工具坐标系和基坐标系。
- 可如下使用工具/基座坐标管理：
 - 在控制系统的 smartPAD 上
(>>> 4.4.1 “smartPAD 上的工具坐标和基坐标管理” 页面 106)
 - 在 WorkVisual 中



- 事后更改工具坐标系和基坐标系可能对已创建的程序或已示教的位置有影响。
- 工具坐标系和基坐标系的更改要与现有的存档或备份合并/相协调。

4.4.1 smartPAD 上的工具坐标和基坐标管理

说明

在 smartPAD 上可以测量和校正工具坐标系和基坐标系。在此也可以新建或删除坐标系。

操作步骤

1. 菜单路径：机器人按键 > 投入运行 > 工具/基坐标管理



图 4-48：菜单路径

2. “工具/基坐标管理”窗口打开。

工具/工件选项卡已预选。

下面的屏幕截图涉及的是 KUKA College 的演示工具。

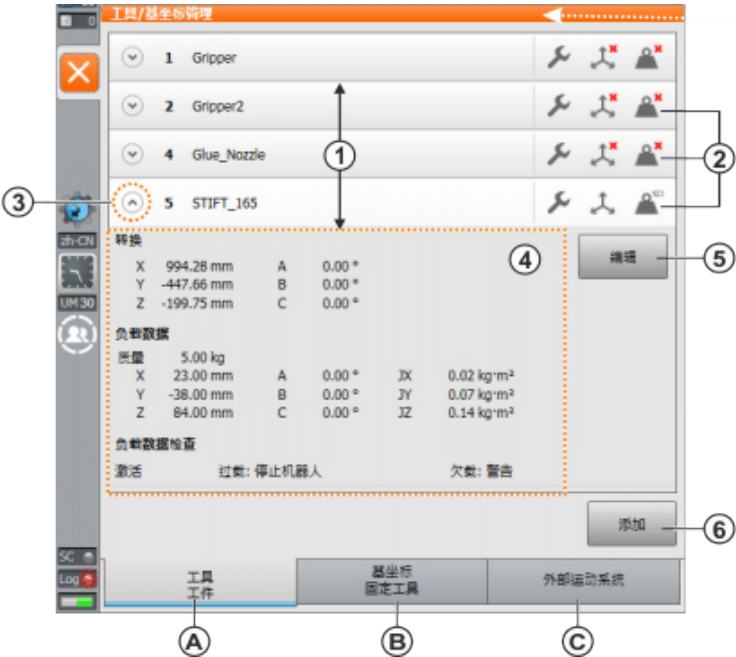


图 4-49：工具/基坐标管理

A	选项卡	工具或工件
B	选项卡	基坐标或固定工具
C	选项卡	外部运动系统
1	控制系统上现有工具或工件的概览	– 工具/工件编号 – 工具/工件名称 – 快速概览，参见第 2 点

2	快速概览	测量方式  = 工具  = 工件 测量坐标系  = 控制系统上的测量 (例如: 4 点法)  = 数字测量  = 无测量, 没有识别到偏移量 负载数据  = 负载数据已保存  = 缺少负载数据
3	图标 	为选择的工具或工件展开已保存的测量数据和负载数据。
4	概览 工具或工件	测量数据和负载数据 转换 负载数据 负载数据检查
5	编辑按钮	打开工具或工件测量的配置窗口
6	添加按钮	创建一个新的工具或工件。

3. 对于**基坐标/固定工具**，切换到同名选项卡（B）中。

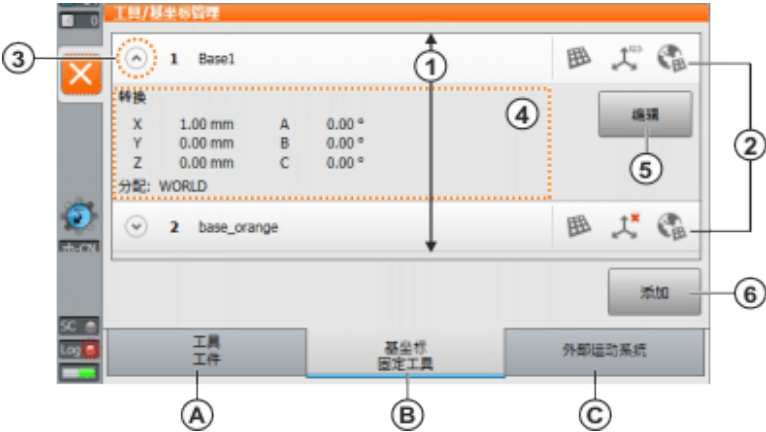


图 4-50：工具/基坐标管理

A	选项卡	工具或工件
B	选项卡	基坐标或固定工具
C	选项卡	外部运动系统
1	控制系统上现有基坐标或固定工具的概览	基坐标或固定工具的编号 基坐标或固定工具的名称 快速概览，参见第 2 点

2	快速概览	– 测量方式  = 固定的工具  = 基坐标系 – 测量坐标系  = 控制系统上的测量（例如：3 点）  = 数字测量  = 无测量，没有识别到偏移量 –  = 参照世界坐标系测量的坐标系
3	图标 	为选择的基坐标或固定工具展开已保存的偏移量数据。
4	概览 工具或工件	基坐标偏移量数据 – 转换 – 分配
5	编辑按键	打开基坐标或固定工具测量的配置窗口。
6	添加按键	创建一个新的基坐标或一个固定工具。

4. 如果要测量**外部运动系统**，则切换到同名选项卡（C）中。

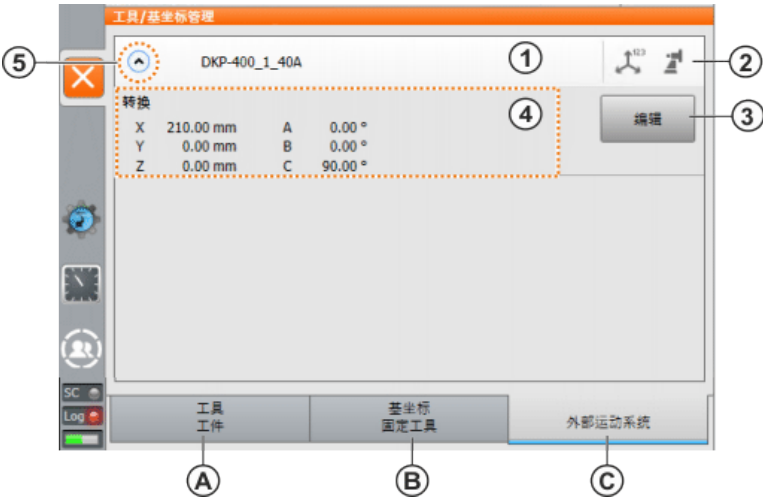


图 4-51：外部运动系统选项卡

A	选项卡	工具或工件
B	选项卡	基坐标或固定工具
C	选项卡	外部运动系统
1	控制系统上现有外部运动系统的概览	– 固定工具的名称

2	快速概览	- 运动系统类型  = 外部运动系统  = 机器人作为 RoboTeam 用户。右侧的编号是 RoboTeam 索引。  = 输送机 - 测量坐标系  = 控制系统上的测量（例如：3 点）  = 数字测量  = 无测量，没有识别到偏移量
3	编辑按钮	打开测量外部运动系统的配置窗口
4	转换	转换基点/坐标原点
5	图标 	为选择的基坐标或固定工具展开已保存的偏移量数据。



无法通过 smartPAD 手动添加外部运动系统。该运动系统必须通过 WorkVisual 集成到当前项目中，可在 smartPAD 上打开和测量。

4.5 机器人上的负载

除了法兰外，在机器人上也可以将负载固定在小臂、大臂和转盘上指定的拾取点。控制系统必须知晓负载和相关的重心距离。

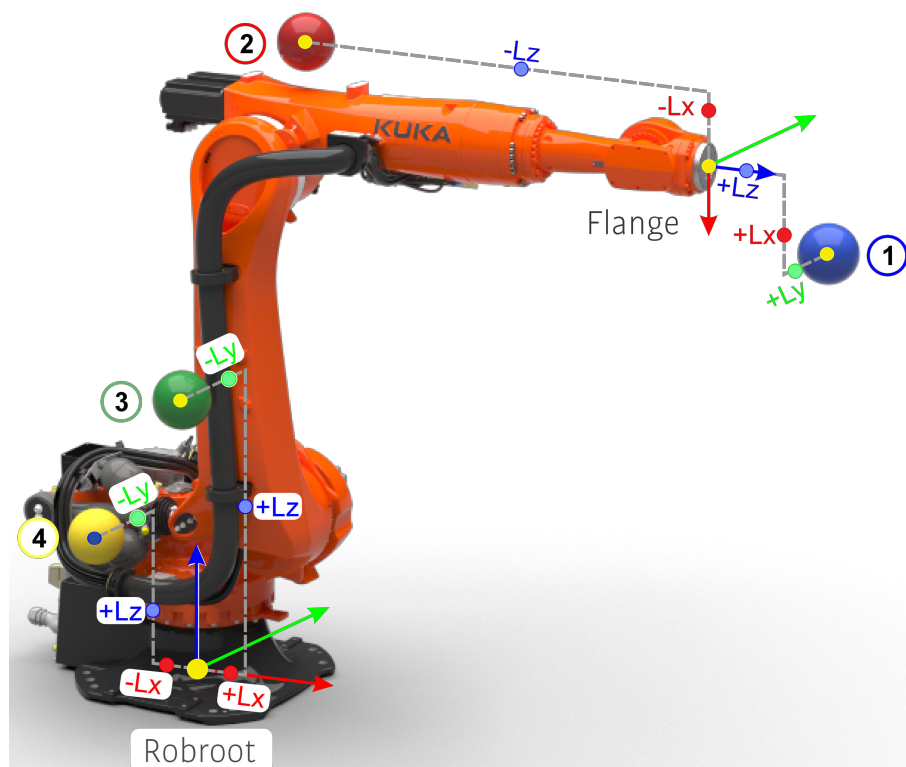


图 4-52: 机器人上的负载

- | | | | |
|---|-----------|---|-----------|
| 1 | 负载能力 | 3 | 轴 2 的附加负载 |
| 2 | 轴 3 的附加负载 | 4 | 轴 1 的附加负载 |

4.5.1 工具负载数据

什么是工具负载数据？

- 工具负载数据是指所有装在机器人法兰上的负载。
- 它是另外装在机器人上并和机器人一起移动的质量。
- 输入的数值为质量、重心位置（质量作用点）和质量惯性矩与相关的主惯性轴（主惯性轴 A、B 和 C 的姿态）。
- 负载数据**必须**输入机器人控制器，并分配给正确的工具。
- **例外：**如果在控制系统上安装了备选软件包 KUKA.LoadDataDetermination，可以在测量期间自动确定负载数据。

工具负载数据的可能来源如下：

- 软件选项 KUKA.LoadDataDetermination（仅用于负载）
- 制造商数据
- 人工计算
- CAD 程序

负载数据的影响

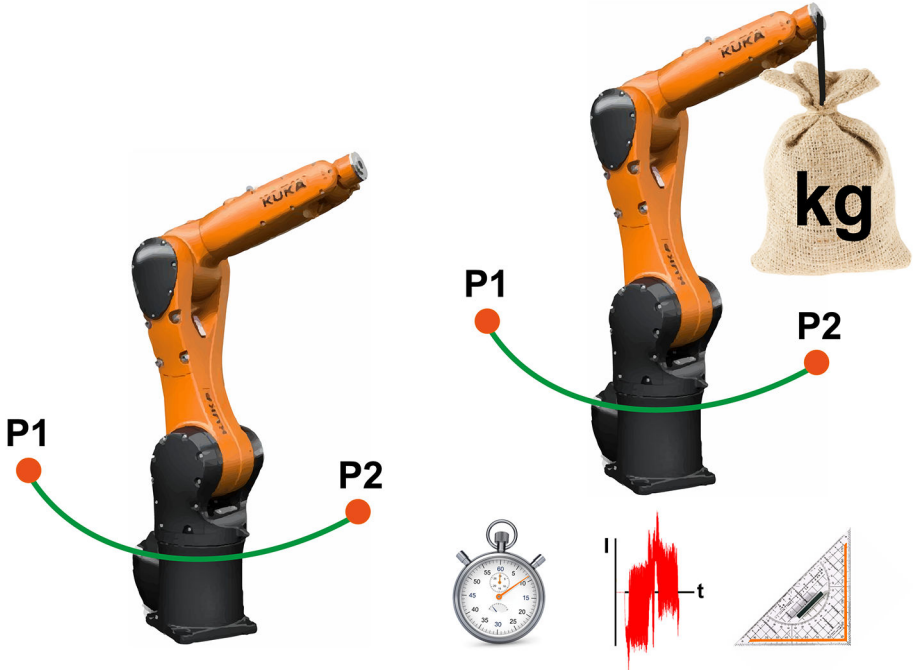


图 4-53：工具负载数据错误的影响

- 输入的工具负载数据会影响许多控制过程。
其中包括：
 - 控制算法（计算加速度）
 - 速度和加速度监控
 - 力矩监控
 - 碰撞监控
 - 能量监控
 - 等等
- 所以，正确输入工具负载数据是非常重要的。
- 如果机器人以正确的工具负载数据执行其运动，则：
 - 可以从它的高精度中受益，
 - 可以使运动过程具有最佳的节拍时间，
 - 可以使机器人达到长的使用寿命（由于磨损小）。

操作步骤

测量工具时介绍了如何输入工具负载数据。
请参见： (>>> [“保存工具/工件负载数据” 页面 123](#))

4. 5. 2 机器人上的附加负载

机器人上的附加负载

附加负载是在转盘、小臂或大臂上附加安装的组件，例如：

- 拖链系统
- 阀门
- 上料系统
- 材料储备

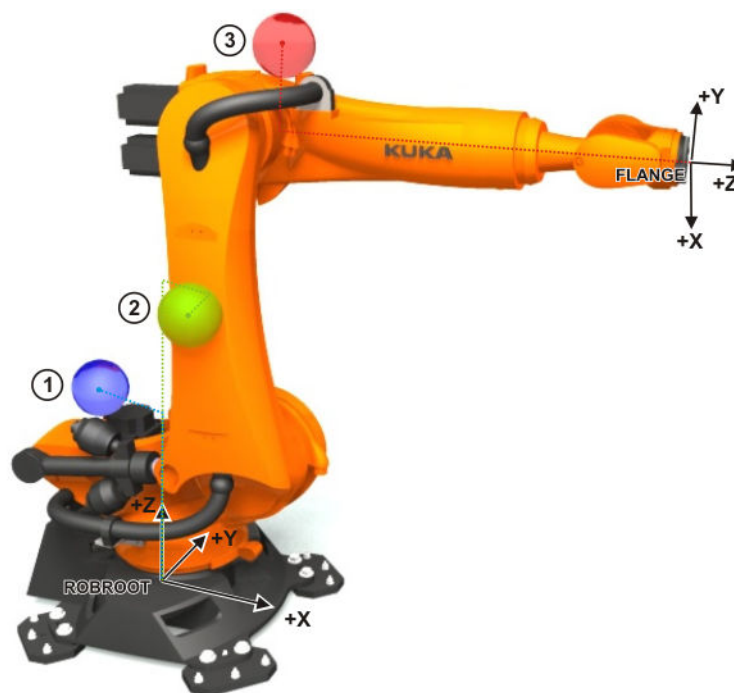


图 4-54：机器人上的附加负载

提示

并不是每个机器人都可以在全部三个位置安装附加负载。

附加负载数据必须输入机器人控制器。

必要的数据包括：

- 质量 (m)，单位：kg
- 物体重心至参照系 (X、Y 和 Z) 的距离，单位：mm
- 主惯性轴与参照系 (A、B 和 C) 的夹角，单位：度 (°)
- 物体绕惯性轴 (J_x 、 J_y 和 J_z) 的转动惯量，单位：kgm²

附加负载的 X、Y、Z 值的参考系：

负载	参考系
附加负载 A1	ROBROOT 坐标系 $A1 = 0^\circ$
附加负载 A2	ROBROOT 坐标系 $A2 = -90^\circ$
附加负载 A3	FLANGE 坐标系 $A4 = 0^\circ$, $A5 = 0^\circ$, $A6 = 0^\circ$

附加负载数据可以参阅下列来源：

- 制造商数据
- 人工计算
- CAD 程序

附加负载对机器人运动的影响

附加负载数据的数值以不同的方式对机器人运动产生影响：

- 轨迹规划
- 加速度

- 节拍时间
- 磨损



警告

如果运行附加负载数据错误的机器人，则会导致人员受伤和生命危险并/或导致严重财产损失。

操作步骤

- 菜单路径：机器人按键 > 投入运行 > 附加负载数据
- 输入固定附加负载（1）的轴的编号。用**继续**键(3)确认。

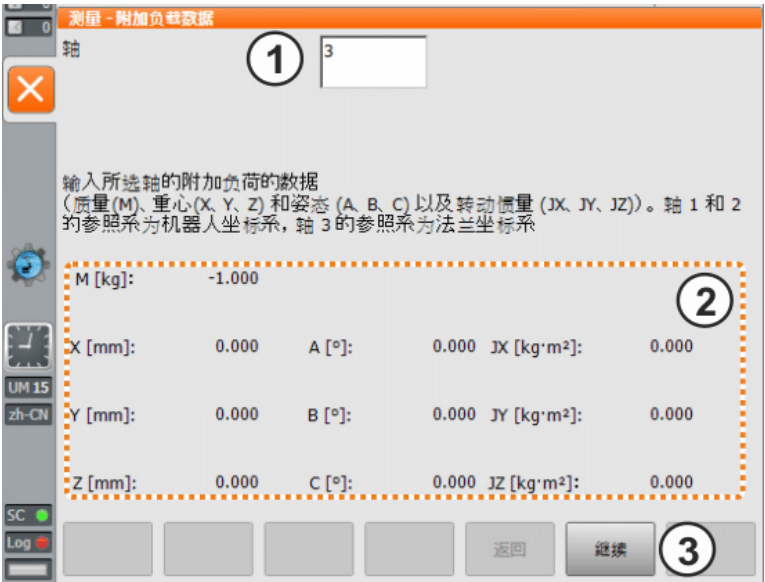


图 4-55：选择负荷的轴

- 输入附加负载数据（1）。用**继续**键(2)确认。

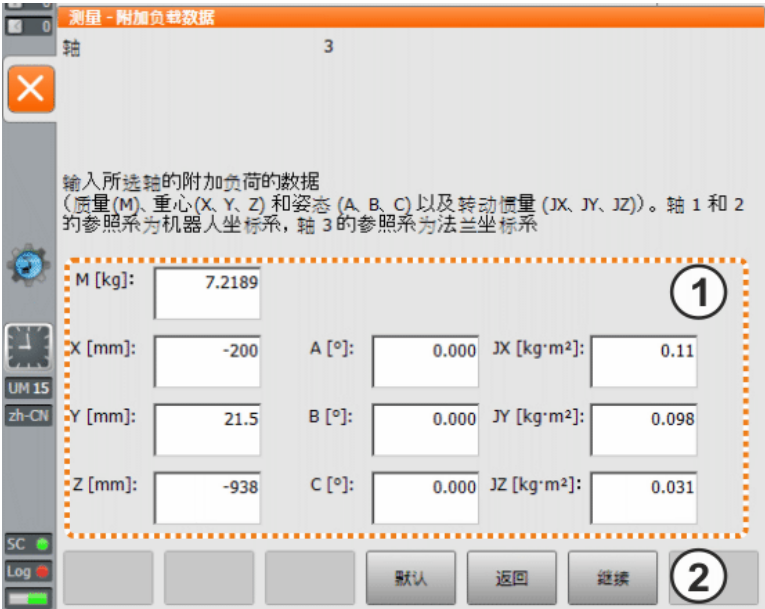


图 4-56：附加负载的数据

- 按下**保存**。

4.6 工具测量

说明

测量工具意味着生成一个以工具参照点为原点的坐标系。该参照点被称为 TCP (Tool Center Point, 即工具中心点), 该坐标系即为工具坐标系。



图 4-57: 测量的工具坐标系示例

- 测量工具意味着生成一个以工具参照点为原点的坐标系。
- 该参照点被称为 TCP (Tool Center Point, 即工具中心点), 该坐标系即为工具坐标系。

测量步骤

测量时, 工具坐标系到法兰坐标系的距离 (用 X、Y 和 Z) 以及之间的转角 (用角度 A、B 和 C) 被保存。

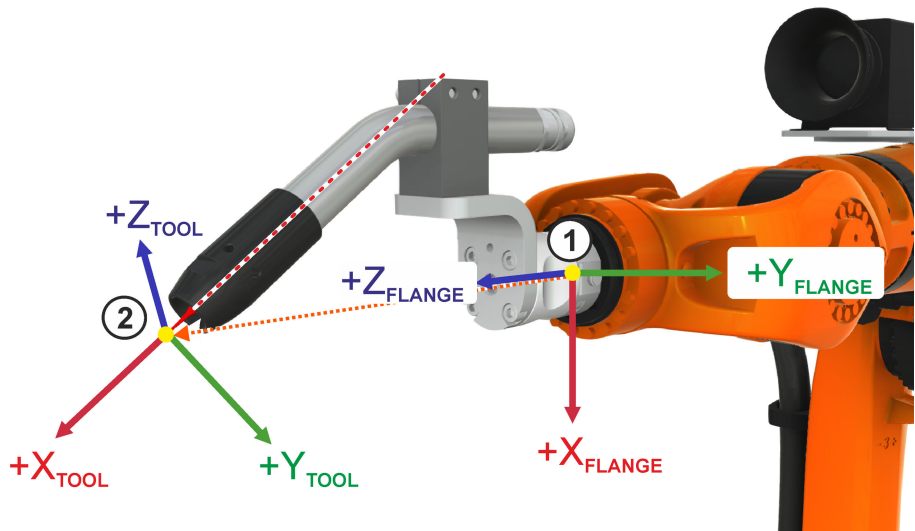


图 4-58：测定工具

- 1 法兰坐标系
- 2 工具坐标系

提示

在标准配置中可以保存 16 个工作坐标系。（变量：TOOL_DATA[1…16]）。

优点

如果一个工具已精确测定，则在实践中对操作和编程人员有以下优点：

绕 TCP 改变姿态

工具可围绕 TCP（例如：工具顶尖）改变姿态/对齐。

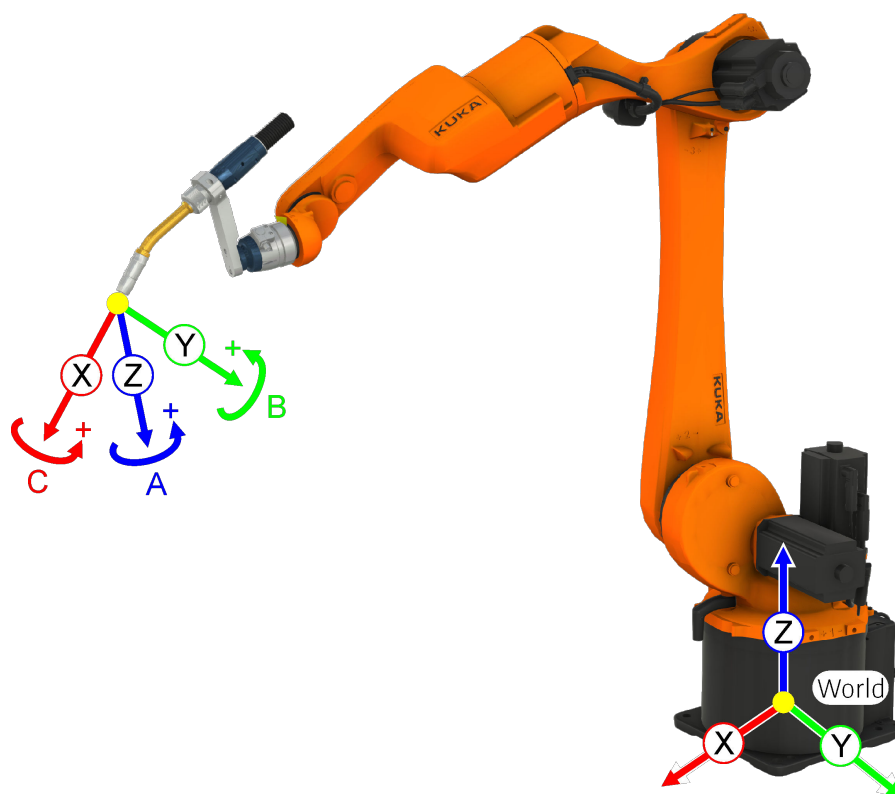


图 4-59：绕 TCP 改变姿态

沿工具作业方向移动

机器人可沿工具作业方向移动。

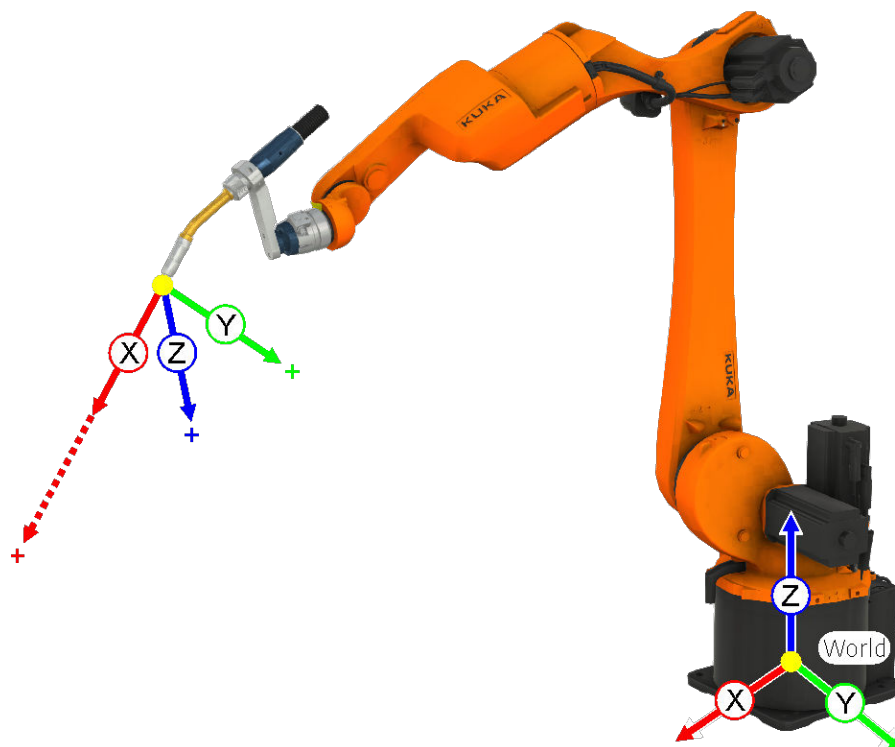


图 4-60：工具工作方向

轨迹运动编程

沿着 TCP 上的轨迹保持已编程的运行速度。

可通过经测量的工具 TCP 沿轨迹进行定义的姿态引导。

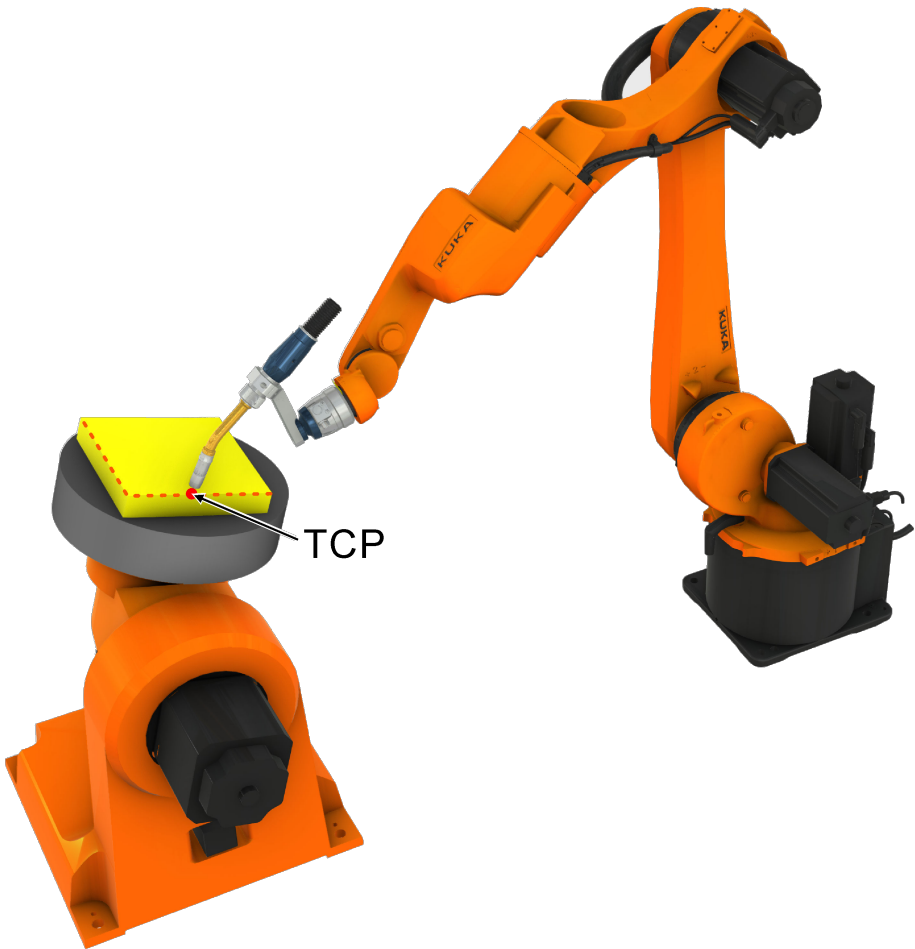


图 4-61：以 TCP 为基准来编程

工具测量的途径

工具测量由三个步骤组成：

步骤	说明
1	确定 TOOL 坐标系统的原点 以下方法可供选择： • <i>XYZ 4 点法</i> • <i>XYZ 参照</i>
2	确定工具坐标系的方向 以下方法可供选择： • <i>ABC-World</i> • <i>ABC 2 点</i>
或者	直接输入至法兰中心点的距离值 (X, Y, Z) 和转角 (A, B, C)。 • <i>数字输入</i>
3	输入工具负载数据 • 工具 + 工件的负载，单位 [kg] • 重心至机器人法兰的距离 • 转动惯量

4.6.1 添加新的工具/工件

操作步骤

添加一个新的工具/工件

1. 在控制系统上打开工具/基坐标管理。
2. 在工具/工件选项卡 (1) 中点击添加按钮 (2)。

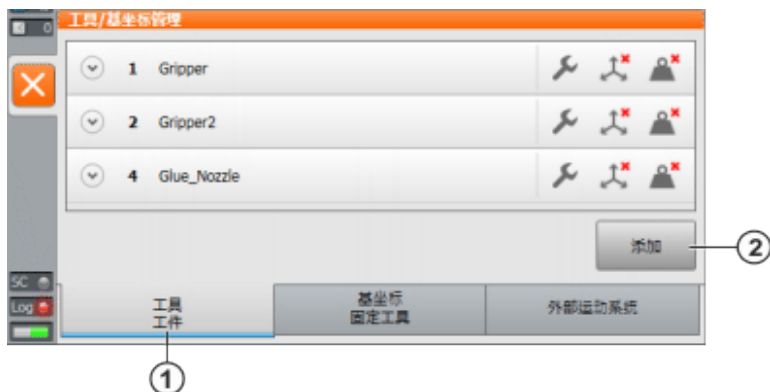


图 4-62: 添加工具/工件

3. 新建工具



图 4-63: 新建工具

- 在下拉窗口 (1) 中选择“未被占用的”工具编号。
- 在输入框 (2) 中为新的工具确定名称。

- 规定待测量对象（3）的类型。
在此选择工具或工件



图 4-64：选择

- 测量方法**
通过点击**测量**按键（4）展开另一个弹出菜单：



图 4-65：测量方法

- 点击所需的测量方法。
确定 TCP
 - XYZ 4 点
(>>> [4.6.1.1 “TCP 测量 4 点法” 页面 120](#))
 - XYZ 参照
(>>> [4.6.1.2 “XYZ 参照法” 页面 124](#))确定方向
 - ABC 世界
(>>> [4.6.1.3 “ABC-World 方法” 页面 127](#))
 - ABC 2 点
(>>> [4.6.1.4 “ABC 2 点法” 页面 130](#))

4.6.1.1 TCP 测量 4 点法

用 XYZ 4 点法进行 TCP 测量

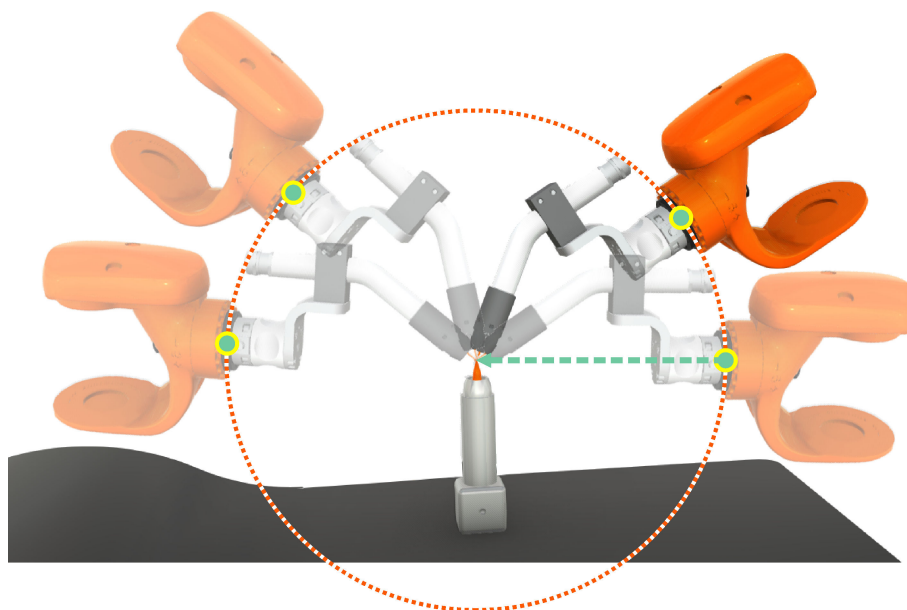


图 4-66: 4 点法

- 将待测量工具的 TCP 从 4 个不同方向移向一个参照点。
- 参照点可以任意选择。
- 机器人控制器从四个储存并测量的法兰位置中计算出 TCP。



移至参照点的四个个法兰位置，彼此必须间隔足够远，并且不得位于同一平面上。

操作步骤

XYZ 4 点法

1. 添加一个新的工具。
(>>> 4.6.1 “添加新的工具/工件” 页面 118)
2. 在机器人的空间附近定位参考点。
3. 选择 XYZ 4 点法进行测量。



图 4-67: 测量方法

4. 在测量菜单中注意工作说明：
将工具从不同方向对准参照点。



图 4-68: XYZ 4 点测量

- 选中校准点/测量点 1。相应的行蓝色标记 (1)。
- 借助 6D 鼠标或点动键将新工具的 TCP 移至参考顶尖。



越精确地驶向这些点，结束时确定的 TCP 越精确。
在相应的位置上降低 H0V 手动倍率或利用增量式手动运行。

- 借助按钮 Touch-Up (2) 保存点坐标。

5. 为剩余的校准点/测量点重复该过程。



图 4-69: XYZ 4 点测量

- 以表格形式显示确定的坐标 (1)。
- 在测量结果栏中显示确定的工具偏移量以及测量误差。



如果测量误差过大，则可以有针对性地重复选中测量点并借助**驶向 (PTP)** 按键 (3) 驶向这些测量点。在该情况下，更精确地驶向该点并重新借助 Touch-Up 按键记录结果。重新检查测量误差。

- **保存** (4) 确定的工具偏移量。

6. 检查并保存工具数据。

The screenshot shows the '工具/坐标系管理 - 编辑工具' (Tool/Coordinate Management - Edit Tool) window. It contains the following sections:

- Header:** 编号 5 (Number 5), 名称 STIFT_165 (Name STIFT_165), 工具 (Tool).
- 转换 (Conversion):** A table for coordinate conversion with columns X, Y, Z, A, B, C. Values are: X=994.28 mm, Y=-447.66 mm, Z=-199.75 mm, A=0.00°, B=0.00°, C=0.00°. A '测量' (Measure) button is on the right. A circled '1' is next to the B and C angle fields.
- 上一次测量 (Last Measurement):** XYZ: XYZ 4 点法 (XYZ: XYZ 4-point method), 误差: 4.89 (Error: 4.89). A '检查并修正' (Check and correct) button is on the right. A circled '2' is next to the error value.
- 负载数据 (Load Data):** 质量 5.00 kg (Mass 5.00 kg). A table for inertia data with columns X, Y, Z, A, B, C, JX, JY, JZ. Values are: X=23.00 mm, Y=-38.00 mm, Z=84.00 mm, A=0.00°, B=0.00°, C=0.00°, JX=0.02 kg·m², JY=0.07 kg·m², JZ=0.14 kg·m².
- 检查负载数据 (Check load data):** 激活 (Activated) checkbox, 过载 停止机器人 (Overload Stop robot), 欠载 警告 (Underload Warning).
- Buttons:** 删除 (Delete), 保存 (Save), 保存 返回 (Save Return), 退出 (Exit). A circled '3' is next to the '保存' (Save) button.

图 4-70: XYZ 4 点测量

- 在**转换**栏 (1) 中检查工具偏移量。
- 在栏 (2) 中显示测量方式以及误差。
如果该工具已测量，则**检查和修正**功能可通过同名按键使用。
- **保存** (3) 所有输入的值和偏移量

保存工具/工件负载数据

操作步骤



图 4-71：负载数据

- 1. 一个工具/工件已定义并且已打开了工具/基坐标管理。
(>>> 4.6.1 “添加新的工具/工件” 页面 118)
- 2. 输入工具负载数据和已有的栏 (1)。



KUKA.LoadDataDetermination 能够自动确定负载数据。如果在控制系统上安装了该备选软件包，则有一个用于执行自动确定的单独按钮可用。

参数/单位		说明
M	kg	质量
X、Y、Z	mm	相对于 FLANGE 坐标系的重心位置
A、B、C	度	惯量主轴相对于 FLANGE 坐标系的姿态
转动惯量：		
JX	kgm ²	绕主轴系统 X 轴的惯量
JY	kgm ²	绕主轴系统 Y 轴的惯量
JZ	kgm ²	绕主轴系统 Z 轴的惯量

- 3. 激活/停用和配置负载数据检查 (3)。

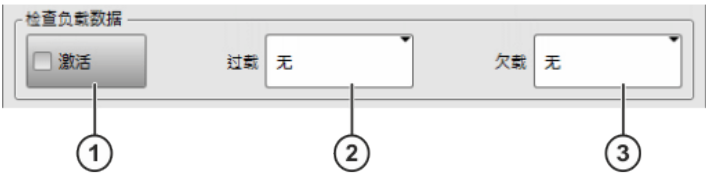


图 4-72：负载数据检查

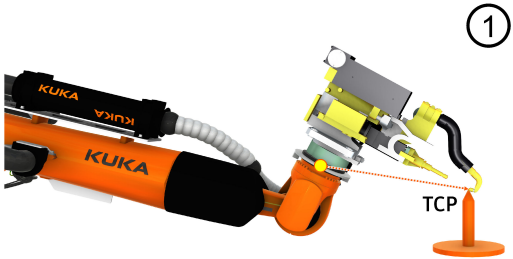
序号	说明
1	勾选：激活对同一窗口中所显示工具的检测。在过载或欠载时则出现规定的反应。 不勾选：不激活对同一窗口中所显示工具的检测。在过载或欠载时不出现反应。
2	在此可以规定过载时应出现何种反应。 – 无：无反应 – 警告：机器人控制器输出以下状态信息：<i>在检查机器人负载（工具 {编号}）时测得过载。</i> – 机器人停机：机器人控制器显示一条内容与警告时相同的但需确认的信息。机器人以 STOP 2 停止。
3	在此可以规定欠载时应出现何种反应。可能的反应与过载时类似。

4.6.1.2 XYZ 参照法

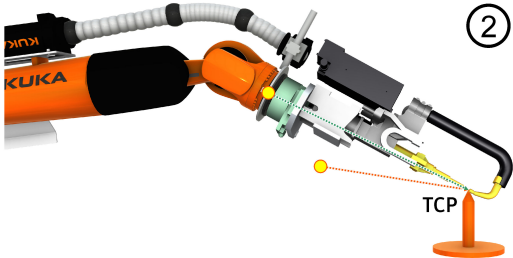
说明

使用 XYZ 参照法，在控制系统中示教具有相似几何形状的多个同类工具。
为此将一个已知的工具（1）用作参照物。机器人控制器比较法兰位置，并计算新工具（2）的 TCP。与 XYZ 4 点法相比，这里的所需测量行驶数量可降低为 2。

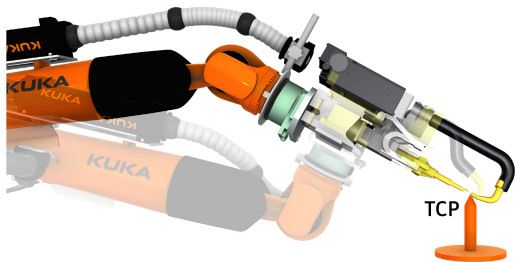
参考工具



新的待测量工具



结果



操作步骤

- 1. 在机器人法兰上固定/更换测量所需的参考工具
- 2. 通过工具和基坐标管理添加了一个新的工具。
(>>> 4.6.1 “添加新的工具/工件” 页面 118)
- 3. 通过测量按键选择 XYZ 参考方法。



图 4-73：测量方法

- 4. 工具和基坐标管理打开。

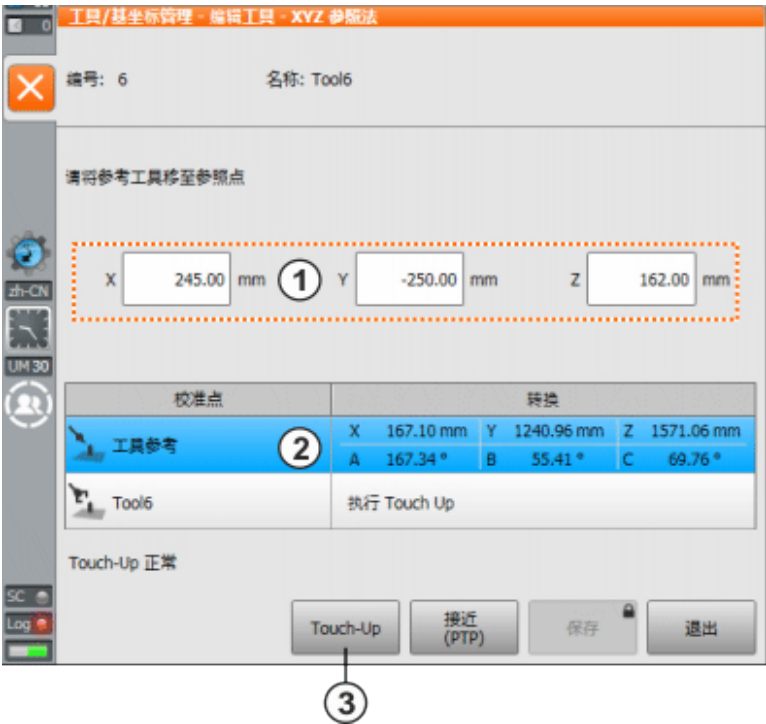


图 4-74：XYZ 参照

- 输入参考工具（1）的偏移量值（TCP）。
为了可以正确地反算为机器人法兰，需要这些值。
如果参考工具本身在控制系统上，则可以从工具和基坐标管理中获取这些数据。
- 点击参考工具行（2）。

该行呈蓝色被选中。

- 将参考工具的 TCP 移到在空间中自由选择的参考点上。
- 借助按键 **Touch-Up** (3) 记录位置。
在转换栏中显示确定的值。



借助**驶向 (PTP)** 按键可以反复驶向已示教过的点。

5. 拆下机器人法兰上的参考工具并用待测量的工具更换。

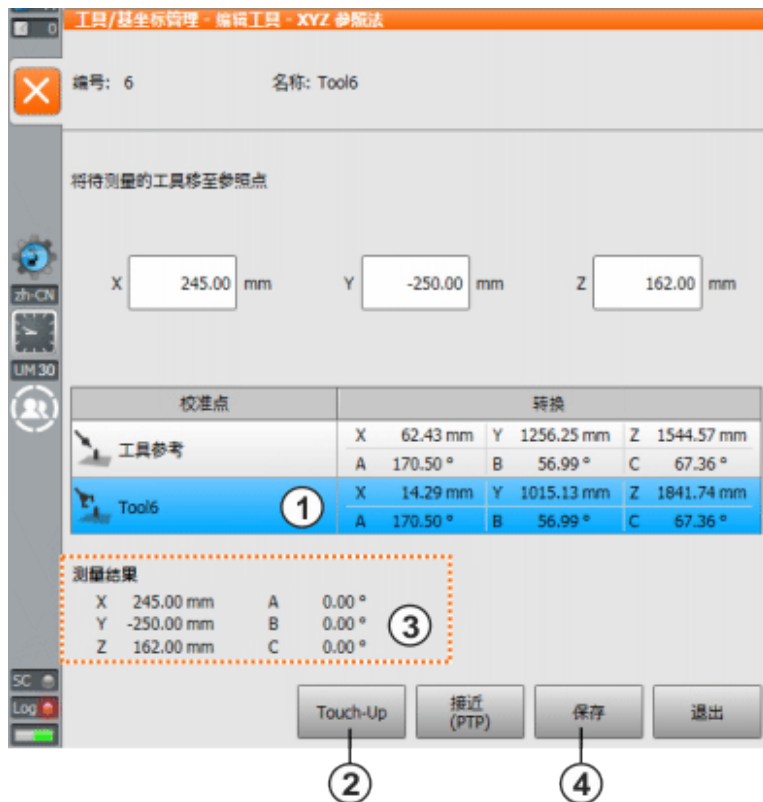


图 4-75: XYZ 参照

- 将工具的 TCP 移到参考点上。
- 点击带**工具名称**的行 (1)。
该行呈蓝色被选中。
- 用按键 **Touch-Up** (2) 应用位置。
在转换栏中显示确定的值。
- 在**测量结果**窗口 (3) 中显示确定的工具偏移量。

6. 用按键 **OK** (4) 备份测量结果。



在工具/基坐标管理的工具概览中可查看参考工具的偏量值。



4.6.1.3 ABC-World 方法

说明

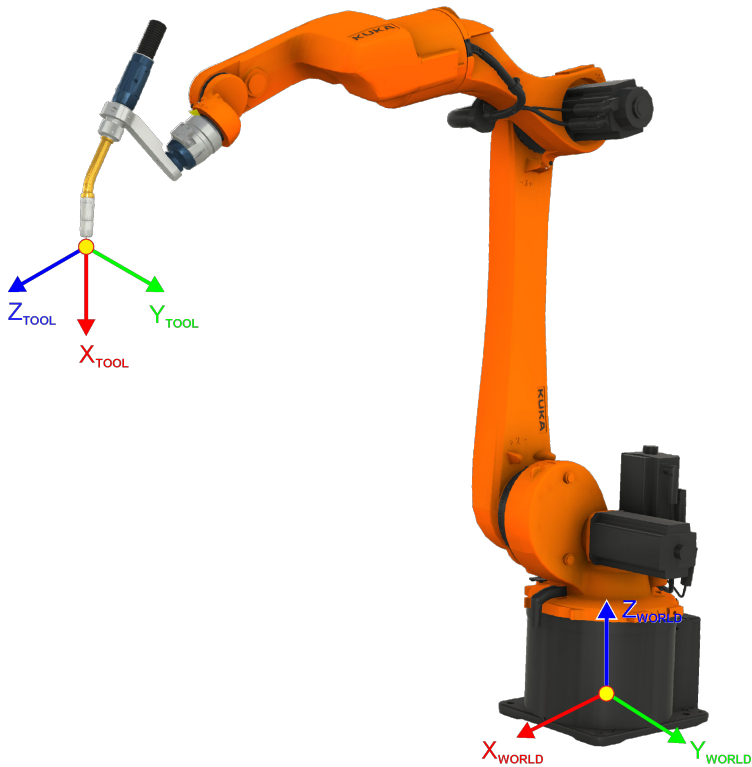


图 4-76: ABC 世界坐标系法

- 将 TOOL 坐标系的轴调整为与 WORLD 坐标系的轴平行。
- 机器人控制器以此得知 TOOL 坐标系的姿态。

类型

此方法有两种变型：

- 5D
 - 可用 5D 类型确定工具的作业方向。
 - 标准设置中，该作业方向为 X 轴。
 - 其它轴的方向由系统确定，对于用户来说不是很容易识别。
 - 应用范围：例如，MIG/MAG 焊接，激光切割或水射流切割
- 6D:
 - 可用 6D 类型确定所有轴方向，而非仅作业方向
 - 应用范围：如焊钳、夹持器或粘胶喷嘴

操作步骤

1. 一个新的工具已测量并且保存在工具和基坐标管理中。
(>>> 4.6.1.1 “TCP 测量 4 点法” 页面 120) 或者
(>>> 4.6.1.2 “XYZ 参照法” 页面 124)
2. 在工具和基坐标管理中打开该工具。
3. 通过按键**测量**选择 *ABC-World* 方法。



图 4-77：测量方法

4. 编辑工具窗口打开。



图 4-78：ABC 世界

- 在 5D/6D 选择窗口 (1) 中确定姿态确定的种类。
- 点击 ABC 姿态行 (2)。
该行的背景色被置为蓝色。
- 参照世界坐标系对齐机器人的工具。
- 借助按键 Touch-Up (3) 应用姿态。
- 借助同名按键保存 (5) 保存值。



借助驶向 (PTP) 按键可以反复驶向已示教过的点。

练习：尖触头工具测量

练习内容

打开课程练习本并完成以下练习：

- 尖触头工具测量

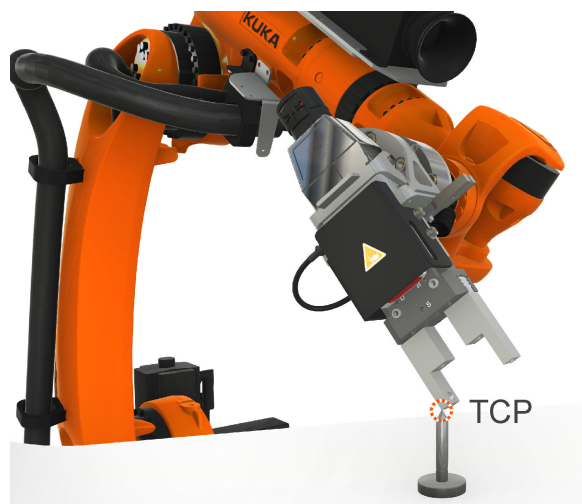
4.6.1.4 ABC 2 点法

操作步骤

通过驶至负 X 轴上的一个点和 XY 平面上的一点，机器人控制器可得知工具坐标系的轴数据。不仅要确定作业方向、还要确定所有轴的方向时，可使用这种方法。

校准点

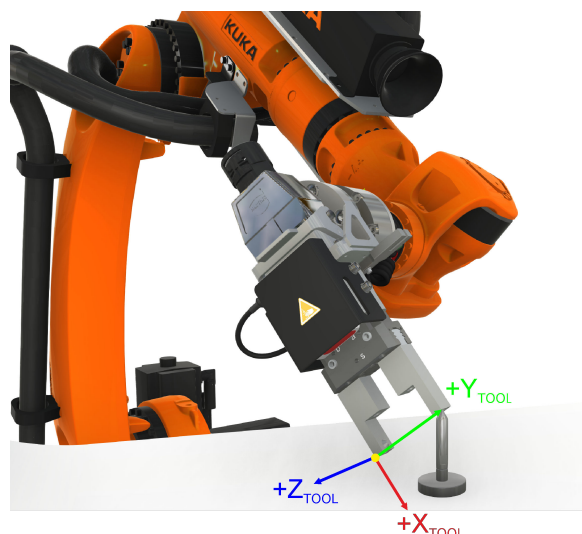
步骤 1:
TCP



步骤 2:
X 轴



步骤 3:
XY 平面



操作步骤

- 1. 一个新的工具已测量并且保存在工具和基坐标管理中。
(>>> 4.6.1.1 “TCP 测量 4 点法” 页面 120) 或者
(>>> 4.6.1.2 “XYZ 参照法” 页面 124)
- 2. 在工具和基坐标管理中打开该工具。
- 3. 通过按键**测量**选择 ABC 2 点 方法。



图 4-79：测量方法

- 4. **编辑工具 - ABC 2 点**窗口打开。
校准点：TCP



图 4-80：校准点，TCP

- 点击 TCP 校准点 (1) 行。
该行的背景色被置为蓝色。
 - 将工具的 TCP 移到确定的参考点上。
 - 借助按键 **Touch-Up** (2) 应用位置。
- 5. **校准点：X 轴和 XY 平面**



图 4-81: ABC 2 点, 测量

X 轴

- 将参考点移向待测量工具负 X 轴（作业方向）上的一个点。
- 点击 X 轴校准点 (2) 行。
该行的背景色被置为蓝色。
- 借助按钮 **Touch-Up** (A) 应用位置。

Y 轴

- 将参考点移向待测量工具 XY 平面上一个具有正 Y 值的点。
- 点击 XY 平面校准点 (3) 行。
该行的背景色被置为蓝色。
- 借助按钮 **Touch-Up** (A) 应用位置。



借助**驶向 (PTP)** 按钮可以反复驶向已示教过的点。

- 在测量结果窗口 (4) 中显示确定的姿态，它表示相对于法兰坐标系的距离和偏转。
- 借助同名按钮**保存**保存该姿态。

练习：抓爪工具测量，2 点法

练习内容

打开课程练习本并完成以下练习：

- 抓爪工具测定，2 点法

4.7 在工具坐标系中移动机器人

在工具坐标系中手动运行

在工具坐标系中手动运行时，可沿着已测量工具的轴线性移动机器人。坐标系并非固定不变（世界坐标系或基坐标系），而是由机器人引导。同时，**所有**机器人轴都移动。工具坐标系的原点被称为 **TCP - Tool Center Point**，并与工具的工作点相对应，例如粘胶喷嘴的尖端。

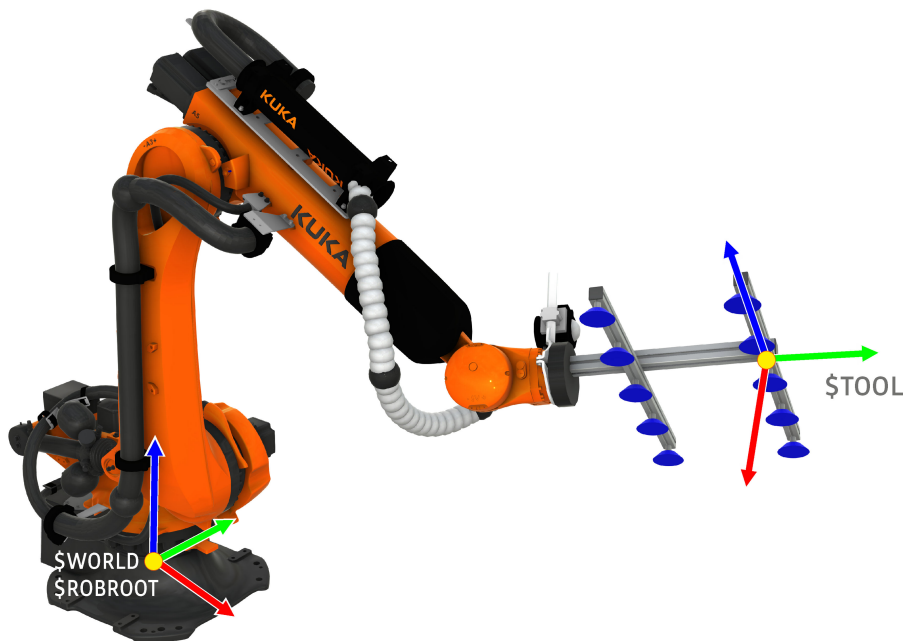
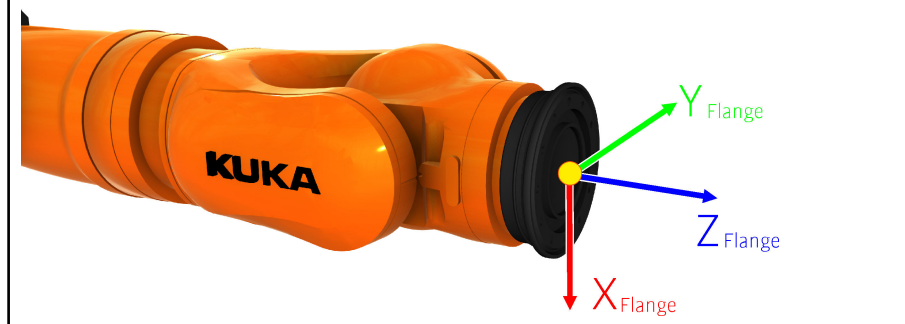


图 4-82：机器人工具坐标系

在 T1 运行模式下手动运行，必须按下使能开关。为此需要使用移动键或者 KUKA smartPAD 的 6D 鼠标。可通过 **HOV 手动倍率**更改速度。在标准配置中有 16 个不同的工具坐标系可供选择。

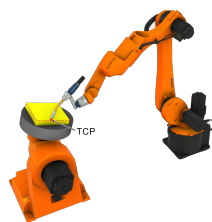


手动运行时，未经测量的工具坐标系始终等于法兰坐标系。



工具坐标时手动运行的原则

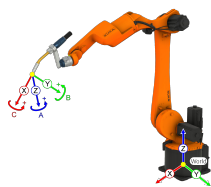
在坐标系中可以两种不同的方式移动机器人：



平移

直线沿坐标轴方向

坐标系：X、Y、Z



转动
环绕着坐标系的坐标轴方向旋转/回转：
角度：A、B 和 C

优点

- 一旦工具坐标系已知，机器人的运动就始终可预测。
- 可以沿工具作业方向移动和绕工具中心点调整方向。
在此过程中必须避免出现奇点位置。



工具作业方向是指工具的工作方向或者加工方向。如果沿 X+ 运行，则工具在工具坐标系中（通常）沿工件方向移动。

工具作业方向的示例有：

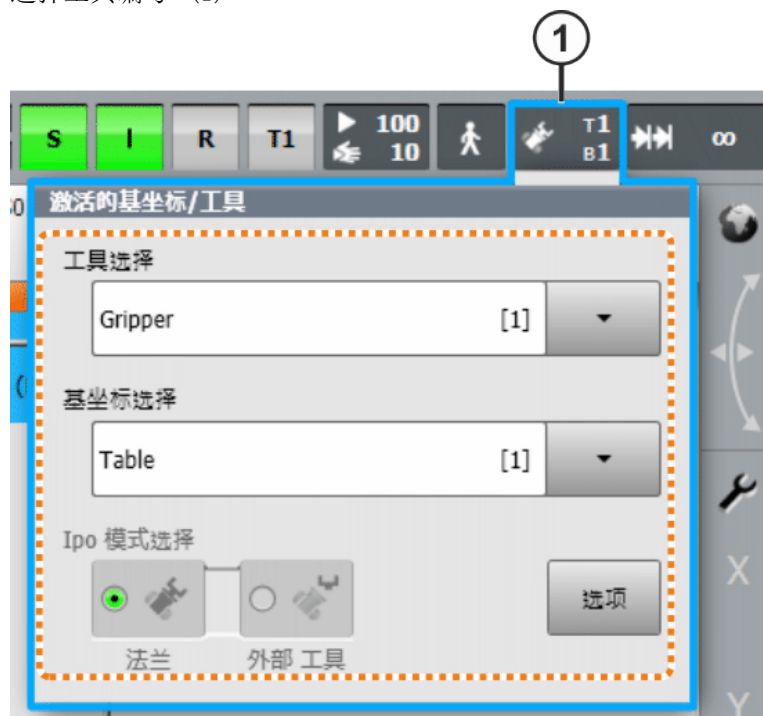
- 粘胶喷嘴上粘接剂的溢出方向
- 搬运时的抓取方向（导向芯棒）
- 电阻点焊的固定电极方向

操作步骤

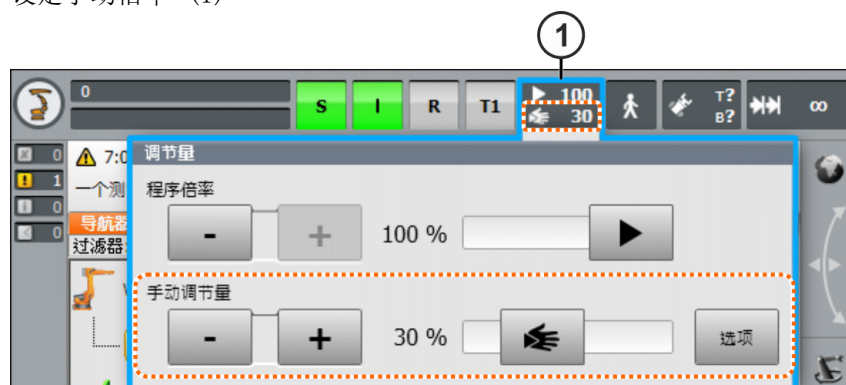
1. 选择工具（1）作为需用的坐标系。



2. 选择工具编号 (1)



3. 设定手动倍率 (1)



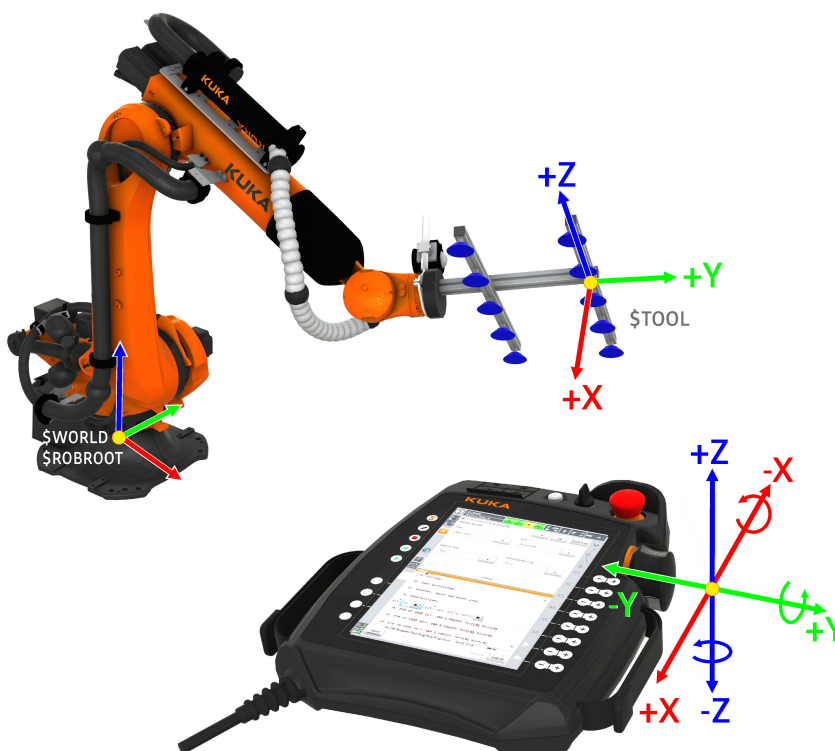
4. 将四个确认键中的一个按至中间位置并按住。



5. 用点动键移动机器人。



6. 可选：用 6D 鼠标将机器人朝所需方向移动。



4.7.1 练习：工具坐标系中的手动运行

练习内容

打开课程练习本并完成以下练习：

- 在工具坐标系中的手动运行

4.8 测量基坐标

说明

基坐标系 *测量* 表示参照世界坐标系在机器人周围的某一个位置上创建坐标系。其目的是使机器人的手动运行以及编程设定的位置均以该坐标系为参照。因此，例如设定的工件支座、抽屉、货盘或机器的边缘均可作为基坐标系合理的参照点。

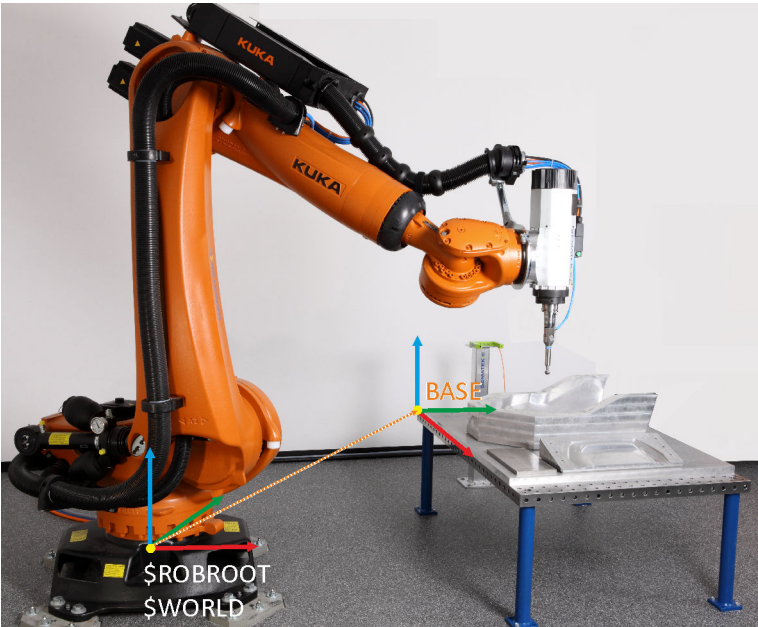


图 4-83：基坐标，使用

初始情况

机器人应从点阵板上拾取 USB 记忆棒进行精加工。在实施程序时应确定一个规定的工具和参考基坐标。这样做具有一些优势。

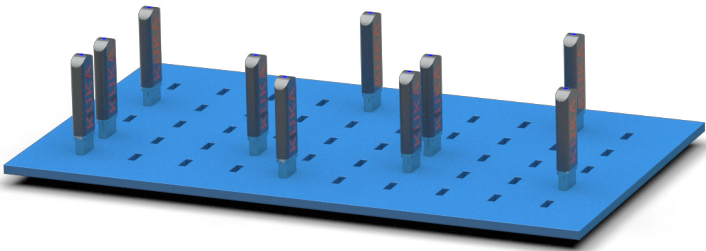


图 4-84：输出示例

优点

基坐标测量后有以下优点：

- 沿着工件边缘移动
可以沿着工件的工作面边缘手动运行 TCP。

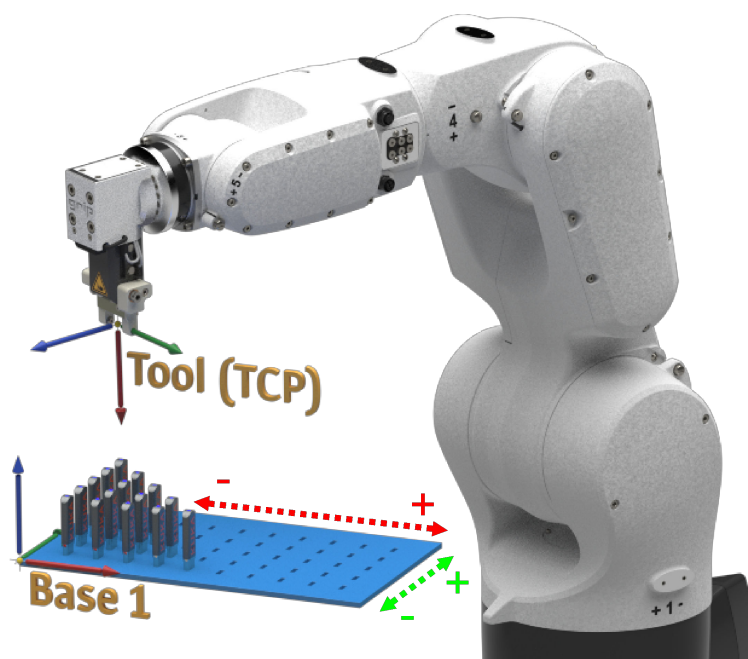


图 4-85：基坐标测量的优点：移动方向

- 参照坐标系：
示教的点以所选的坐标系为参照。

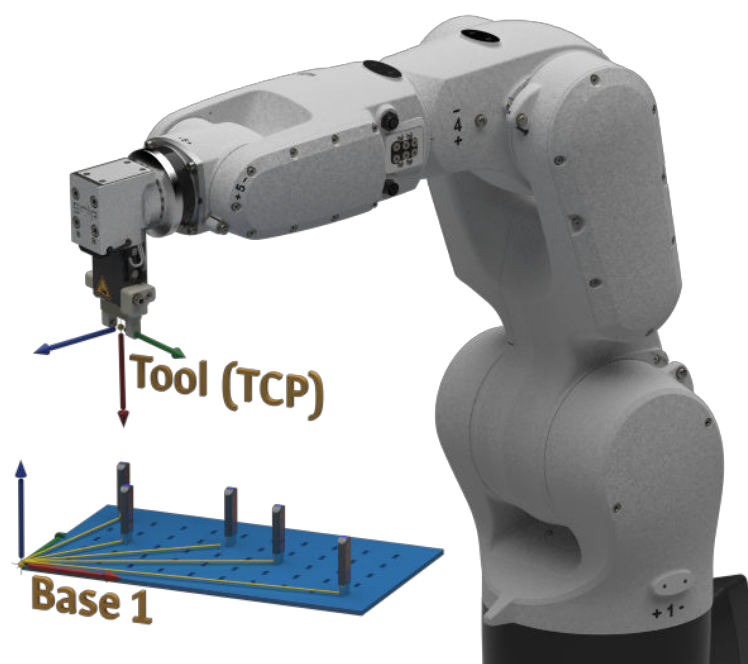


图 4-86：基坐标测量的优点：以所需坐标系为参照

- 坐标系的修正/推移：
可以参照基坐标对这些点示教。如果必须推移基坐标，例如由于工作面被移动，这些点也随之移动，不必重新进行示教。

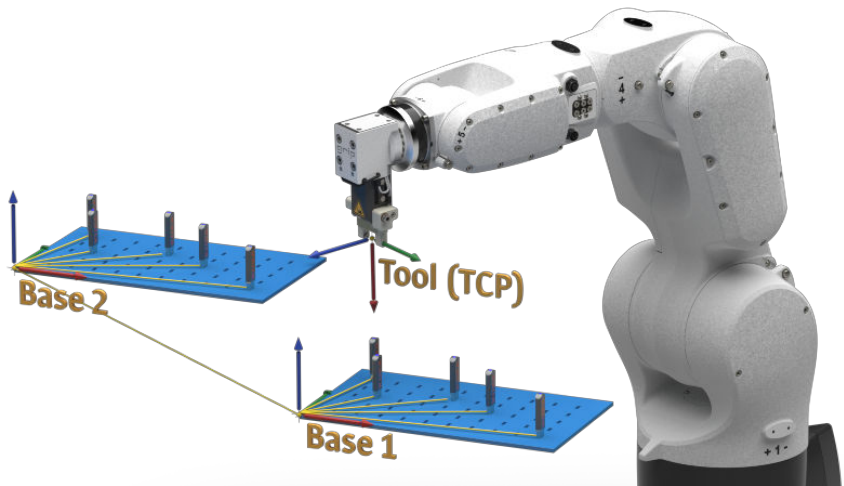


图 4-87：基坐标测量的优点：基坐标系的位移

- 多个基坐标系的益处：
最多可建立 32 个不同的坐标系，并根据程序步加以应用。

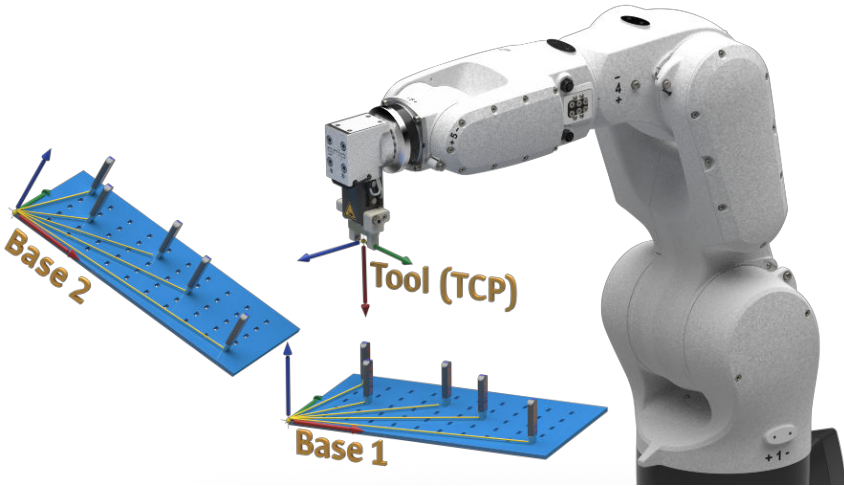


图 4-88：基坐标测量的优点：使用多个基坐标系

基坐标测量的方法

以下的基坐标测量方法可供使用：

方法	说明
3 点法	1. 定义原点 2. 定义 X 轴正方向 3. 定义 Y 轴正方向 (XY 平面) (>>> 4.8.1.1 “三点法” 页面 143)
间接法	• 当无法逼近基座原点时，例如，由于该点位于工件内部，或位于机器人工作空间之外时，须采用间接法。 • 此时须逼近 4 个相对于待测量的基坐标其坐标值 (CAD 数据) 已知的点。机器人控制系统根据这些点计算基座。
数字输入	• 直接输入至世界坐标系的距离 (X、Y、Z) 和转角 (A、B、C) 。

4.8.1 添加新的基坐标/固定工具

操作步骤

添加一个新的基坐标/固定工具

- 1. 在控制系统上打开工具/基坐标管理。
- 2. 在基坐标/固定工具选项卡 (1) 中点击添加按钮 (2)。

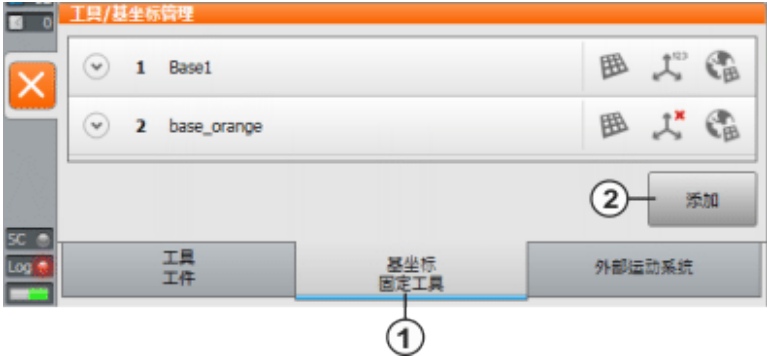


图 4-89：添加基坐标

编辑新建的基坐标/固定工具



图 4-90：编辑基坐标

- 1. 为基坐标/固定工具选择/输入所需的存储位置编号 (1) 和名称 (2) 所用坐标系编号将自动隐藏。因此无法覆盖现有的基坐标。
- 2. 在下拉窗口 (3) 中确定待测量的坐标系是基坐标还是固定工具。
- 3. 借助同名按钮保存 (4) 保存在此之前输入的数据。



可随时保存当前状态。

- 4. 如果基坐标/固定工具相对于世界坐标系的偏移量数据已知，则可在转换栏中用数字直接输入这些数据。
- 5. 如果偏移量数据未知，则也可以通过测量技术确定。为此，点击测量按钮 (5)。

测量方法

通过点击按钮**测量**展开另一个弹出菜单：



图 4-91：测量基坐标，选择

点击所需的测量方法。

- 3 点
(>>> [4.8.1.1 “三点法” 页面 143](#))
- 间接

4.8.1.1 三点法

操作步骤



基坐标测量只能用一个事先已测定的工具进行（TCP 必须已知）。

1. 通过工具和基坐标管理添加了一个新的基坐标。
(>>> [4.8.1 “添加新的基坐标/固定工具” 页面 142](#))
2. 通过按钮**测量**选择 3 点方法。



图 4-92：测量基坐标，选择

3. 用 TCP 接近坐标系的原点。

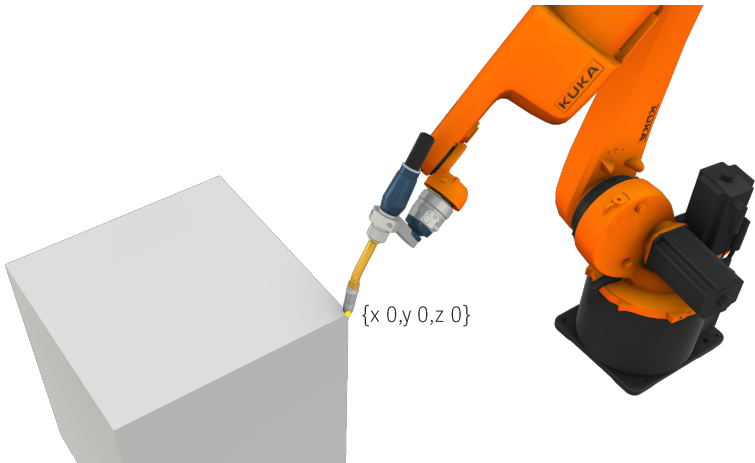


图 4-93：原点的第一个点

4. Touch-Up 原点



图 4-94：原点

- 在下拉菜单 (1) 中选择用于测量的参考工具。
 - 点击原点行 (2)。
该行呈蓝色被选中。
 - 借助按键 Touch-Up (3) 记录位置。
在转换栏中显示确定的值。
5. 用 TCP 接近基坐标系正 X 轴上的一点。

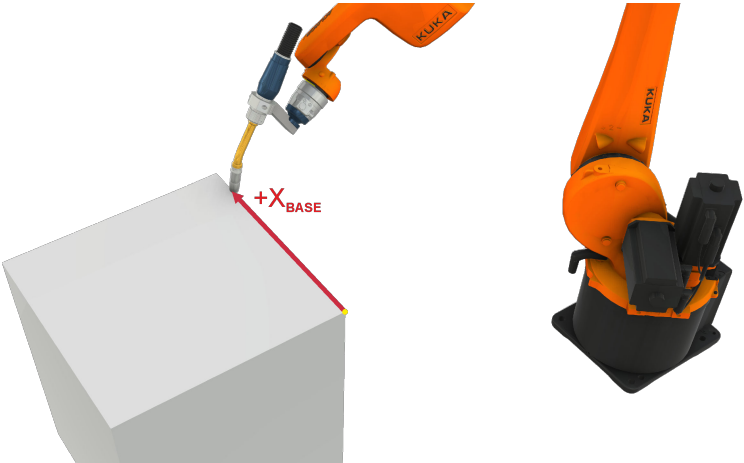


图 4-95：X 方向的第二个点

6. X 轴的 Touch-Up



图 4-96: X 轴

- 点击 X 轴行 (2)。
该行呈蓝色被选中。
 - 借助按键 Touch-Up (3) 记录位置。
在转换栏中显示确定的值。
7. 用 TCP 在 XY 平面上接近带正 Y 值的一点。

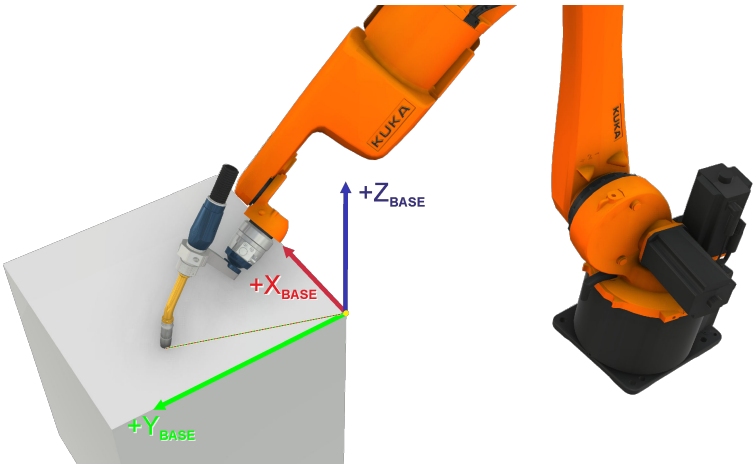


图 4-97: XY 平面上的第三个点

8. XY 平面 Touch-Up



图 4-98: XY 平面

- 点击 **XY 平面**行 (2)。该行呈蓝色被选中。
- 借助按键 **Touch-Up** (2) 记录位置。在转换栏中显示确定的值。
- 基坐标的测量通过记录最后一个点完成。在**测量结果**栏 (3) 中显示偏移量值，它表示相对于世界坐标系的距离和偏转。
- 用同名按键**保存** (4) 保存偏移量值。

练习：工作台的基坐标测量，3 点法

练习内容

打开课程练习本并完成以下练习：

- 工作台的基坐标测量，3 点法

4.8.2 在基坐标系中点动机器人

在基坐标系中运动

机器人的工具可以根据基坐标系的坐标方向运动。基坐标系可以被单个测量，并可以沿工件边缘、工件支座或者货盘调整姿态。由此可以比较方便地进行手动运行。在此过程中，**所有**机器人轴也会自行移动。哪些轴会自行移动，由系统决定，并因运动情况不同而异。在标准配置中有 32 个基坐标系可供选择。

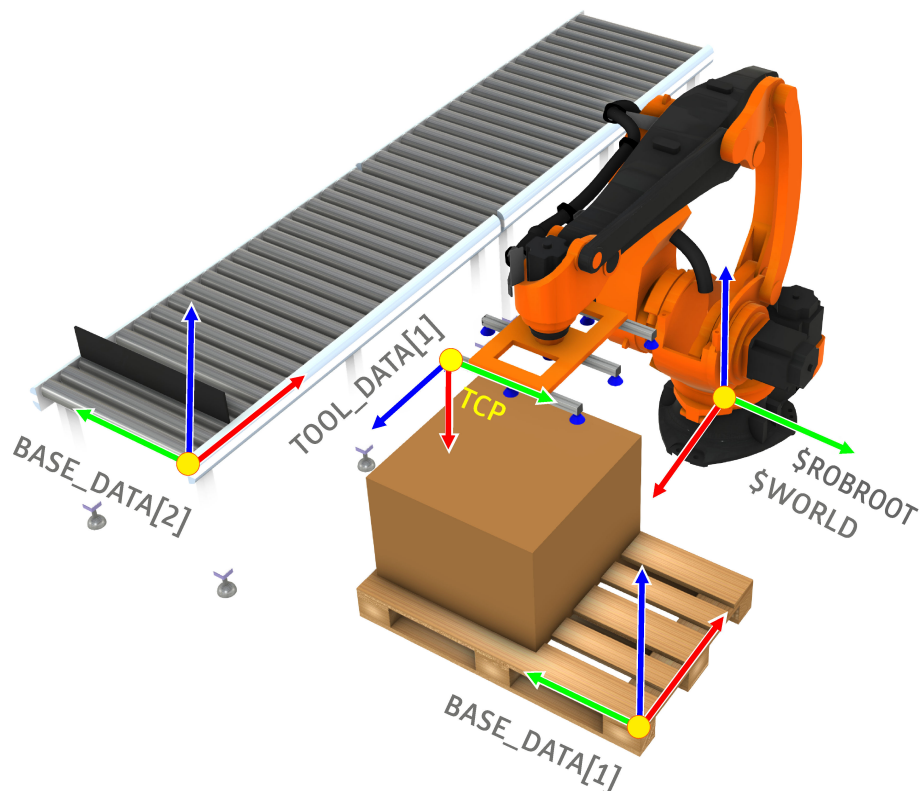


图 4-99：在基坐标系中点动机器人

优点

- 只要基坐标系已知，机器人的运动始终可预测。
- 这里也可用 6D 鼠标直观操作。前提条件是操作员必须相对基坐标系正确站立。如果需要这样使用，则选择 X 轴作为参照。
- 可以平行移动，例如平行于一个工件或托盘棱边。
在此过程中必须避免出现奇点位置。



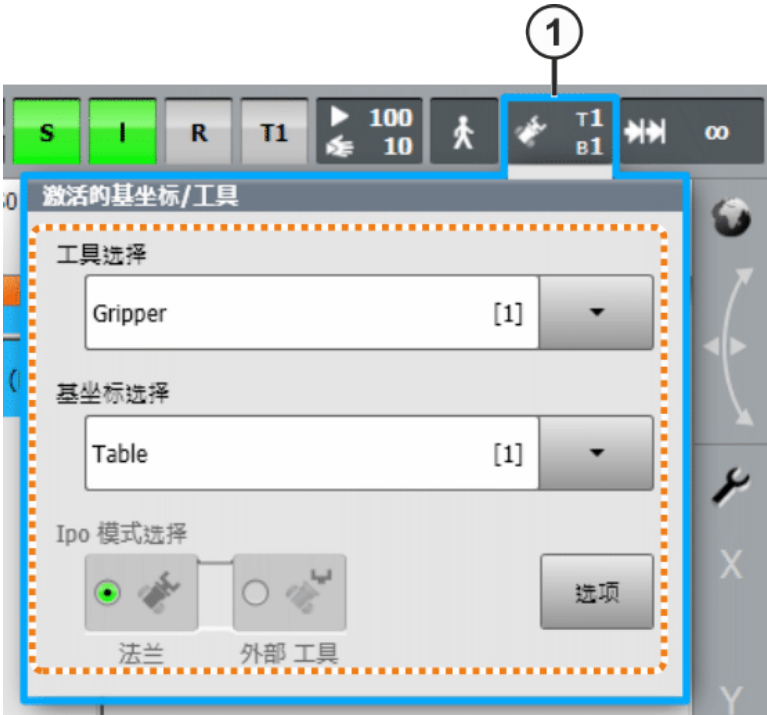
如果还另外设定了正确的工具坐标系，则可在基坐标系中绕工具中心点改变姿态。

操作步骤

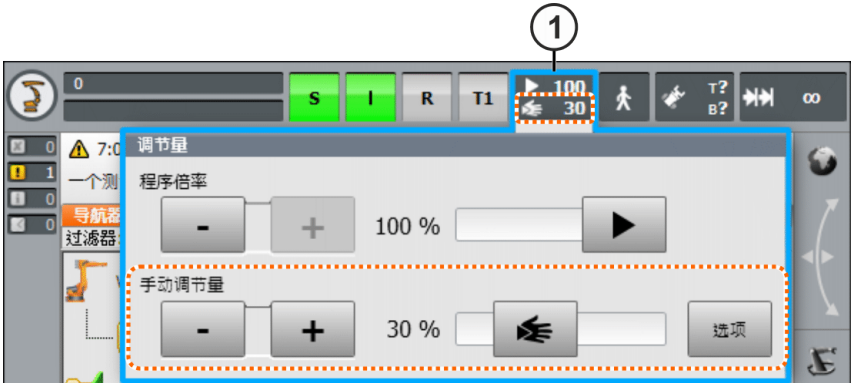
1. 将**基坐标**（1）选为点动键的选项。



2. 选择工具坐标和基坐标（1）



3. 设定手动倍率（1）



4. 将四个确认键中的一个按至中间位置并按住。



5. 用点动键沿所需的方向移动



6. 作为选项，也可用 6D 鼠标来移动。

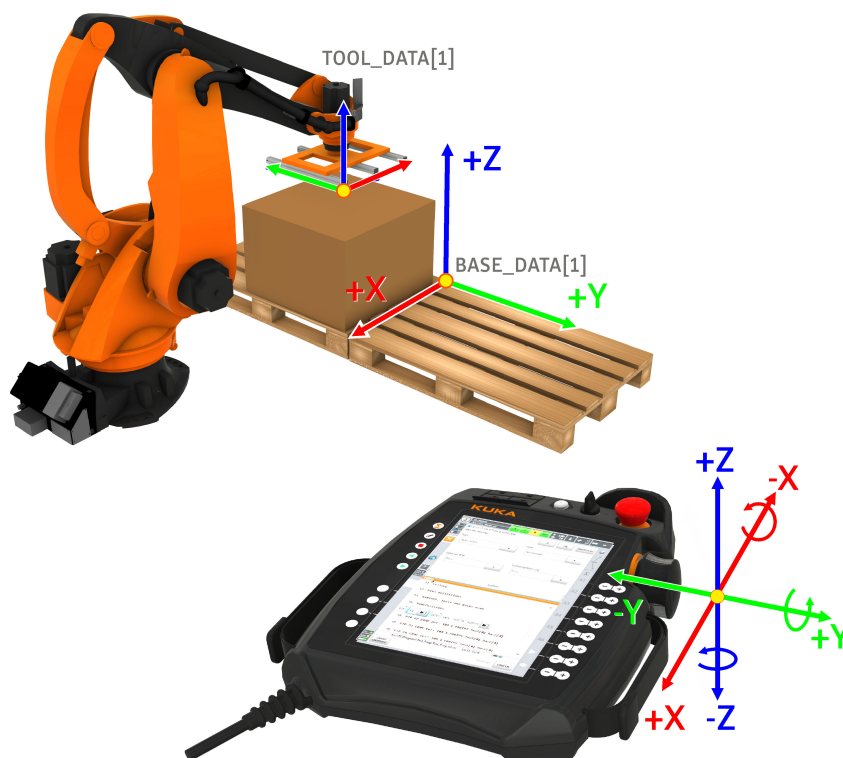


图 4-100: 鼠标移动, 基坐标

4.8.2.1 练习：基坐标系中的手动运行

练习内容

打开课程练习本并完成以下练习：

- 基坐标系中的手动运行

4.9 借助对齐运行方式移动机器人

说明

对齐能够更简单快速地定位机器人。

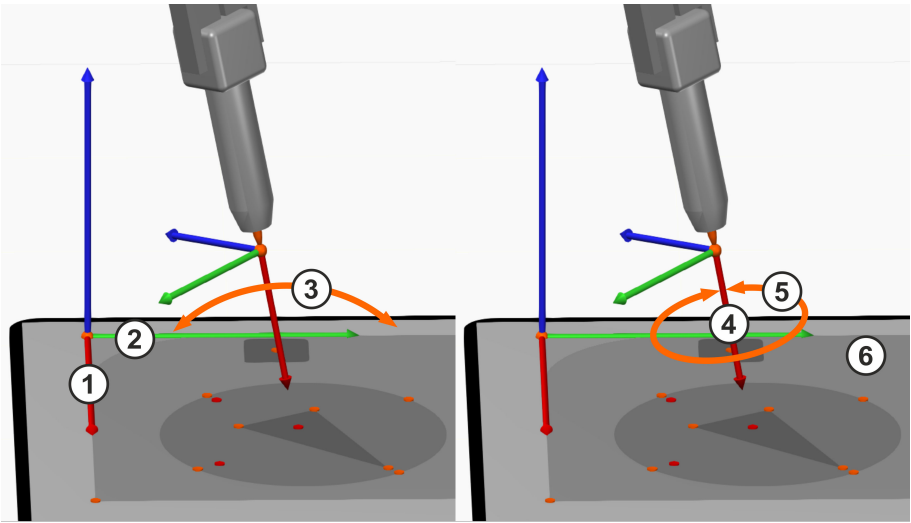


图 4-101：移动机器人：校准

红 X 轴
色
绿 Y 轴
色
蓝 Z 轴
色

- 1. 2 在与对齐方向垂直的使用中的基坐标平面中移动。示例中为 X (1) 或 Y。
- 3 与使用中的基坐标垂直对齐。
- 4 沿使用中的工具的作业方向 (+X) 移动。
- 5 围绕使用中的工具的作业方向转动。
- 6 围绕与基坐标对齐的位置转动（圆锥运动）。



该方法以使用中的工具和基坐标为基准。

操作步骤



只能在 T1 运行模式下使用对齐运行方式。

- 1. 在**手动移动选项**窗口的**按键**选项卡中选择**对齐**选项 (1)。

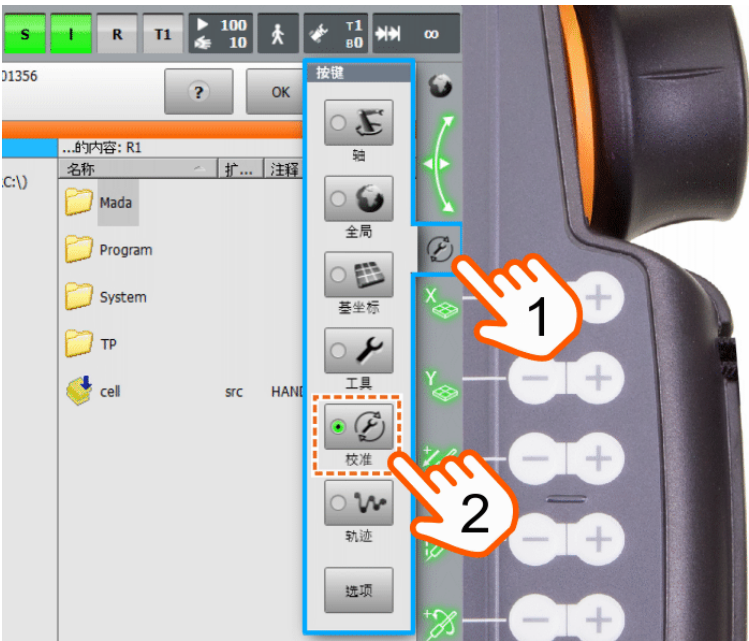


图 4-102：对齐运行方式

2. 设定手动倍率。

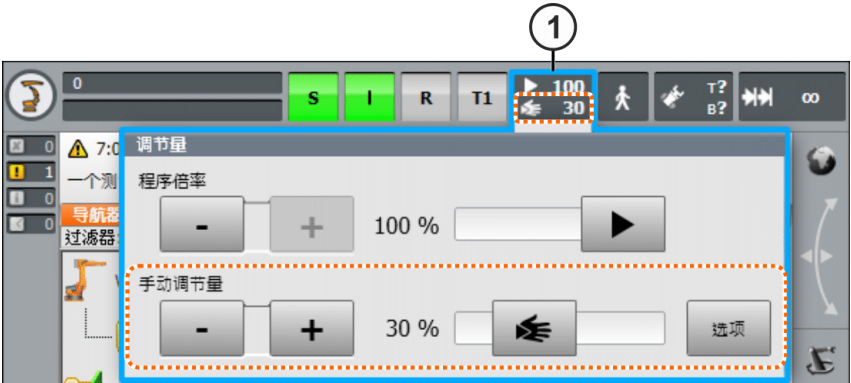


图 4-103：通过状态栏设定手动倍率

3. 按下一个使能开关并按住。



图 4-104：使能开关

4. 现在，借助移动键可以实现以下操作：



图 4-105：手动运行对齐

- 借助移动键（1）可在当前基坐标中沿着显示的方向（如 X）移动
- 借助移动键（2）可在当前基坐标中沿着显示的方向（如 Y）移动

- 借助移动键 (3) 可沿着当前工具坐标的作业方向移动。
- 借助移动键 (4) 可使当前工具与所选基坐标平面垂直对齐。
- 借助移动键 (5) 可将当前工具转动角度 C。
- 借助移动键 (6) 可使当前工具以所选基坐标平面为基准进行圆锥运动。
- 借助基坐标配置 (7) 可确定在基坐标 (1)、(2) 中手动运行的方向，以及用于对齐工具 (4) 的平面。

5. 配置对齐

4. 10 显示内存使用情况

说明

由于并行过程和附加选项，对控制系统内存和硬盘容量的需求增高。该插件提供了当前内存和硬盘使用情况的概览。

操作步骤

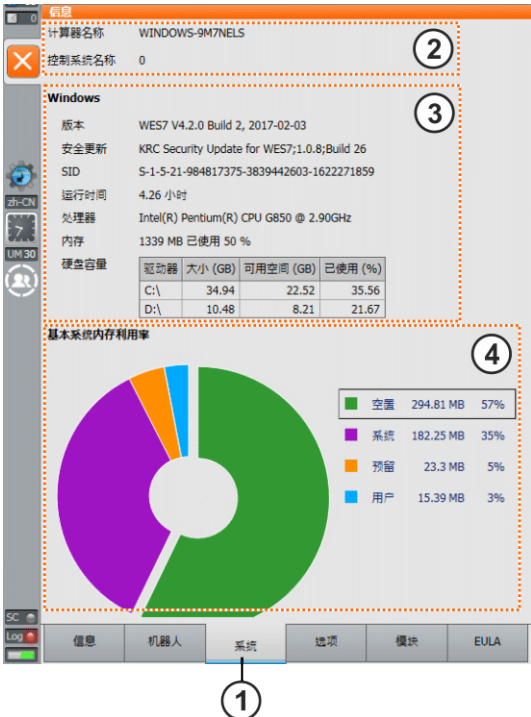


图 4-106：基本系统，内存使用情况

菜单路径：机器人按键 > 帮助 > 信息 > 系统选项卡 (1)

- 计算机名称 (2)
该名称在投入运行期间确定，相当于 Windows 名称。
文件路径：C:\KUKA\WES7RenameComputer
- 控制系统名称 (2)
该名称在投入运行期间设定。
菜单路径：机器人按键 > 投入运行 > 机器人数据
在 RDC 的 EDS 上保存控制系统名称。
- Windows 版本 (3)
已安装 Windows 主系统的版本信息
- SID - 安全识别号 (3)
Windows 主系统的唯一识别
在投入运行期间通过清理步骤 Sysprep 生成 SID。

- **运行时间 (3)**
显示控制系统接通多长时间的运行时间 [h]。
- **处理器 (3)**
所安装处理器的准确型号名称。
- **内存 (3)**
显示可用内存 [MB] 和其利用率 [%]，减去为主系统的预留区域。
- **硬盘大小 (3)**
显示分区 C:\ 和 D:\ 的容量大小和占用情况。
- **主系统的内存使用情况 (4)**
主系统中预留给存储器区域的利用率概览 [MB] [%]。

4.11 控制系统关机

说明

除了主开关外，还可以通过菜单项将控制系统关机或重新启动。该方法实现了情况控制的关机，尤其是在配置、投入运行或故障查找时需要该关机功能。



图 4-107：电力



由于权限管理，大多数情况下只能由特定用户组访问关机菜单项。

提示

主开关过早确认会损坏系统文件

如果重启尚未完成期间选择选项 **重新启动控制系统 PC** 并且按下主开关，就会损坏系统文件。

- 等至重启结束。然后先按下主开关。

操作步骤

1. **菜单路径：** *机器人按键* > *关闭*
2. 选择所需的选项。
需要特殊的用户权限。
3. 按下**关闭控制系统 PC** 或者**重新启动控制系统 PC** 。
4. 用**是**确认安全询问。该系统程序即结束，然后视所选择的选项而定又重新启动。

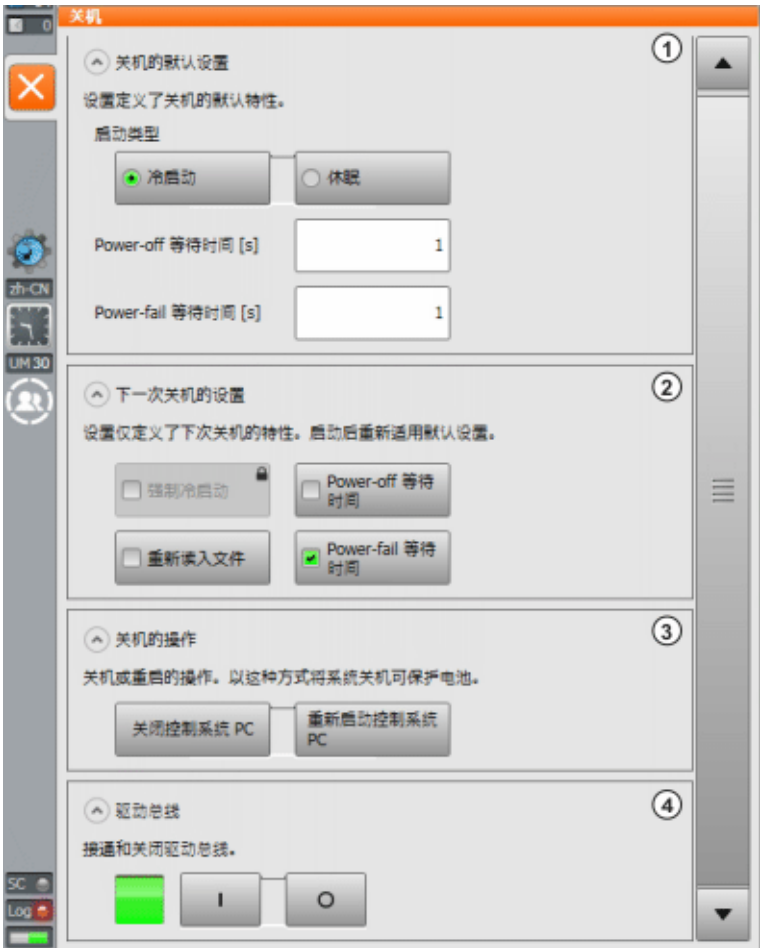


图 4-108：关机

序号	说明
1	关机的默认设置 设置定义了关机的默认特性。 冷启动 冷启动之后机器人控制系统显示导航器。没有选定任何程序。机器人控制系统将重新初始化，例如，所有的用户输出端均被置为 FALSE。要一起考虑到已更改的 XML 文件。 休眠 以休眠方式启动后可以继续执行先前选定的机器人程序。主系统的状态，例如程序、语句显示器、变量内容和输出端，均全部得以恢复。 Power-off 等待时间 [s] 如果控制系统通过主开关关断，则该系统在此处确定的等待时间结束后才关闭。 Power-fail 等待时间 [s] 对于由不可靠电源供电的设备，在断电时抑制控制系统关机的时间，单位：[s]。

序号	说明
2	下一次关机的设置 <i>设置仅定义了下次关机的特性。</i> <i>启动后重新适用默认设置。</i>
	• 强制冷启动 机器人控制系统关机，然后以冷启动方式重新启动。 • 重新读入文件 – 如果通过例如项目激活、网络配置更改等更改了控制系统上的 XML 文件，则必须激活该设置。在大多数情况下，在配置菜单中将出现一条相应的提示。 – 只有在选择了冷启动或强制冷启动时才可用。 • Power-off 等待时间 激活在 (1) 下指定的 Power-off 等待时间 • Power-fail 等待时间 激活在 (1) 下指定的 Power-fail 等待时间
3	关机的操作 <i>关机或重启的操作。以这种方式将系统关机可保护电池。</i>
	• 控制系统 PC 关机 机器人控制系统关机。主开关位于接通位置上并且 CSP 上的休眠 LED 闪烁！ • 重新启动控制系统 PC 在没有按下主开关的情况下，机器人控制系统关机，然后以冷启动方式重新启动。
4	驱动总线 <i>接通和关闭驱动总线</i>
	接通 «» 关闭 • 绿色：驱动总线接通。 • 红色：驱动总线关闭。 • 灰色：驱动总线状态未知。

5 执行机器人程序

5.1 学习单元：执行机器人程序

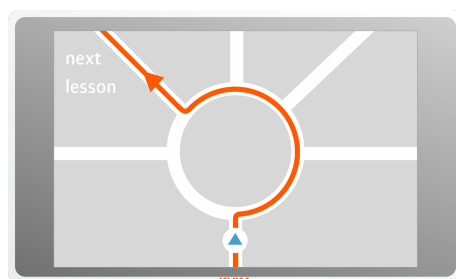


图 5-1：导航器

本学习单元讲授以下内容：

- 选择机器人程序
- 机器人程序看起来是怎样的？
- 程序初始化
- 启动程序

5.2 选择机器人程序

选择和启动机器人程序

如果要执行一个机器人程序，则必须事先将其选中。机器人程序在用户操作界面的导航器中可供使用。运动程序通常保存在文件夹中。Cell 程序（由 PLC 控制机器人的管理程序）始终在文件夹“R1”中。

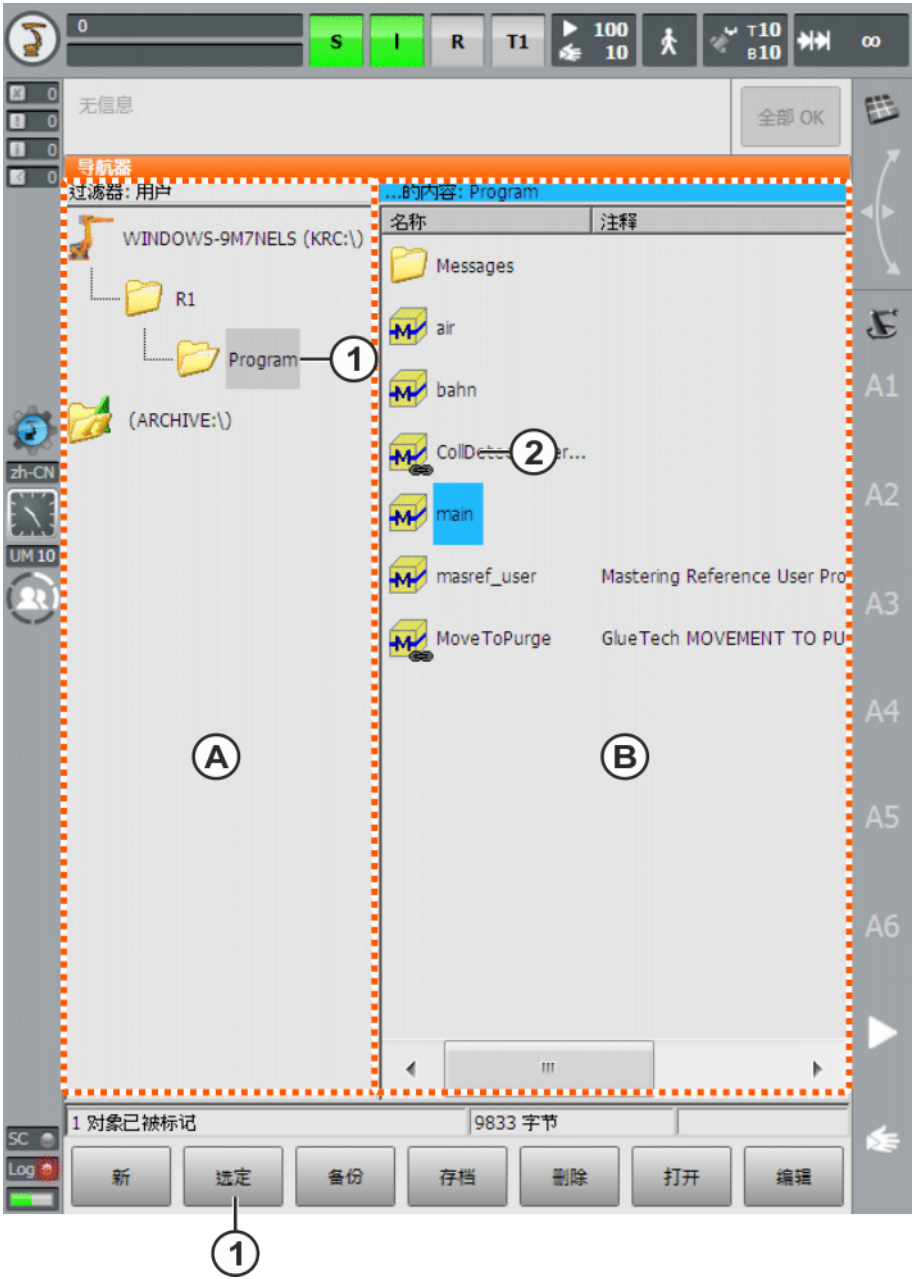


图 5-2：导航器

- A 导航器：文件夹/硬盘结构
- B 导航器：文件夹/数据列表
- 1 选中的程序文件夹
- 2 选中的程序
- 3 用于选择程序的按钮

为启动程序，有启动正向运行程序按钮  和启动反向运行程序按钮  供选择。



- 只有当运行完程序的 INI 行且机器人在此之前刚刚沿待退回的轨迹向前运行后，才能允许用  按键向后运行程序。
 - 但是例如在一次程序停止后手动运行机器人，则无法再进行反向运行。
 - 软件版本 KSS 8.5 及以上，可以使用专门的轨迹“手动运行方式”。(>>> 3.7 “借助轨迹运行方式移动机器人” 页面 70)
- 在此，虽然在轨迹上进行视觉/物理移动，但是已经离开了编程的轨迹。



图 5-3：程序执行方向：向前/向后

序号	说明
1	正向运行启动键
2	反向运行启动键

如果运行某个程序，则对于程序控制的机器人运动，可提供多种**程序运行方式**：

	GO - 程序连续运行，直至程序结尾。 - 在测试模式下必须按住启动键。
	运动 - 每个运动指令单个执行 - 每一个运动结束后，都必须重新按下“启动”键。
	单个步骤 仅供用户组“专家”使用！ - 逐行执行（与行中的内容无关）。 - 每行执行后，都必须重新按下启动键。

5.3 机器人程序看起来是怎样的？

说明

```
1 DEF KUKA_Prog()  
2  
3 INI  
4 SPTP HOME VEL= 100 % DEFAULT  
5 SPTP P1 VEL= 100 % PDAT1 Tool[5] Base[10]  
6 SPTP P2 VEL= 100 % PDAT1 Tool[5] Base[10]  
7 SLIN P3 VEL= 1 m/s CPDAT1 Tool[5] Base[10]  
8 SLIN P4 VEL= 1 m/s CPDAT2 Tool[5] Base[10]  
9 SPTP P5 VEL= 100% PDAT1 Tool[5] Base[10]  
10 SPTP HOME VEL= 100 % DEFAULT
```

11
12 END

行	说明
1 和 12	仅限于在专家用户组中可见： “DEF 程序名()”始终出现在程序开头 “END”表示程序结束
3	“INI”行包含程序正确运行所需的标准参数的调用。 “INI”行必须最先运行！
4 至 10	- 真正的程序文本，包括运动指令、等待/逻辑指令等。 - 行驶指令“SPTP Home”常用于程序开头和末尾，因为这是唯一的已知位置。

5.4 程序初始化

BC0 运行



图 5-4: Sprinter

KUKA 机器人的初始化运行称为 BC0 运行。

BC0 是 **Blockcoincidence**（即程序段重合）的缩写。重合意为“一致”及“时间/空间事件的会合”（即回到程序轨迹上）。

在下列情况下要进行 BC0 运行：

- 选择程序
- 程序复位
- 程序执行时手动运行
- 在程序中更改
- 语句行选择

示例

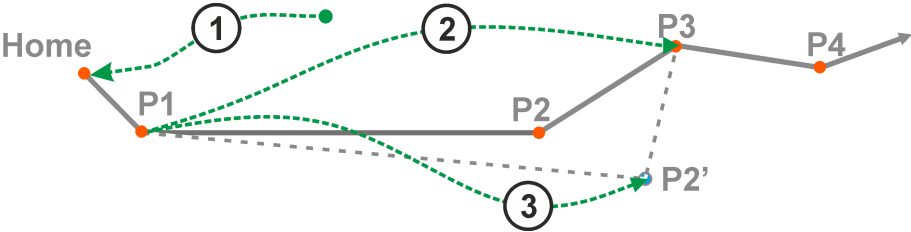


图 5-5: BC0 运行的原因举例

- 1 在选择或者复位程序后 BCO 运行至 Home 位置
图中示例：机器人在初始位置之外 » 程序启动
- 2 语句行选择后执行 BCO 运动。
图中示例：机器人在 P1 中 » 在 P3 上选择语句 » 程序启动
- 3 更改了运动指令后执行 BCO 运行
图中示例：机器人在 P1 中 » 将位置 P2 改为 P2' » 程序启动

BCO 运行的原因

- 为了使当前的机器人位置与机器人程序中当前点的坐标保持一致，必须执行 BCO 运行。
- 仅当当前的机器人位置与编程设定的位置相同时才可进行轨迹规划。因此，首先必须将 TCP 置于轨迹上。

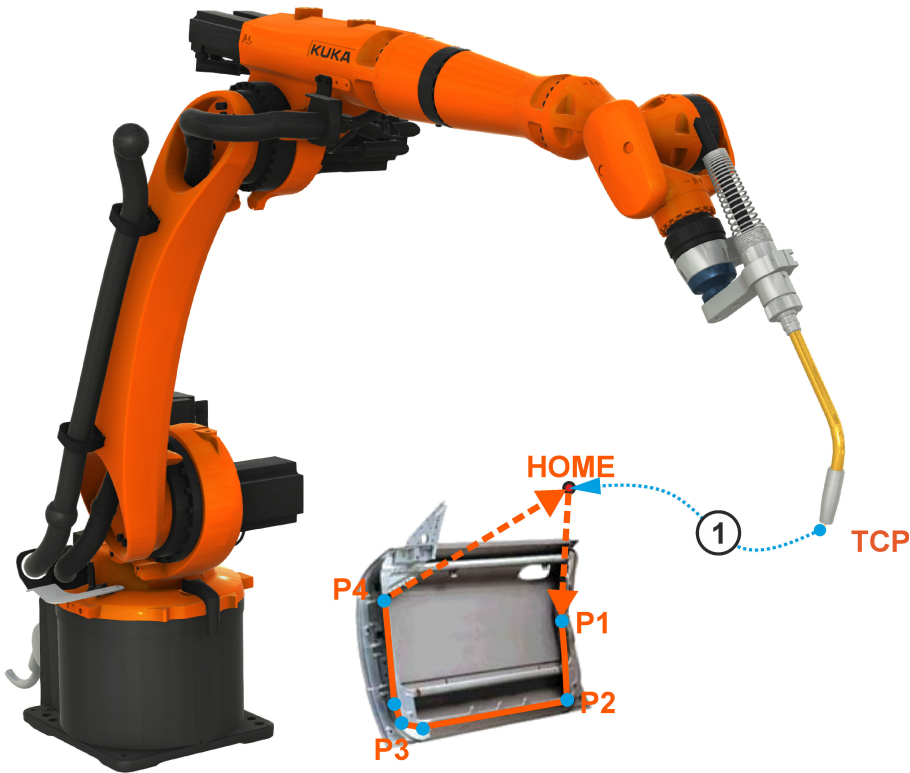


图 5-6：在程序选择时 BCO 运行范例

1	在选择程序或重置程序后 BCO 运行至初始位置/HOME 位置	
KUKA System Software		HOME
Volkswagen System Software		初始位置
提示		
机器人直接（最快轨迹）从当前位置移至初始位置/HOME 位置。碰撞危险！		

5.5 进行程序启动

1. 选择程序

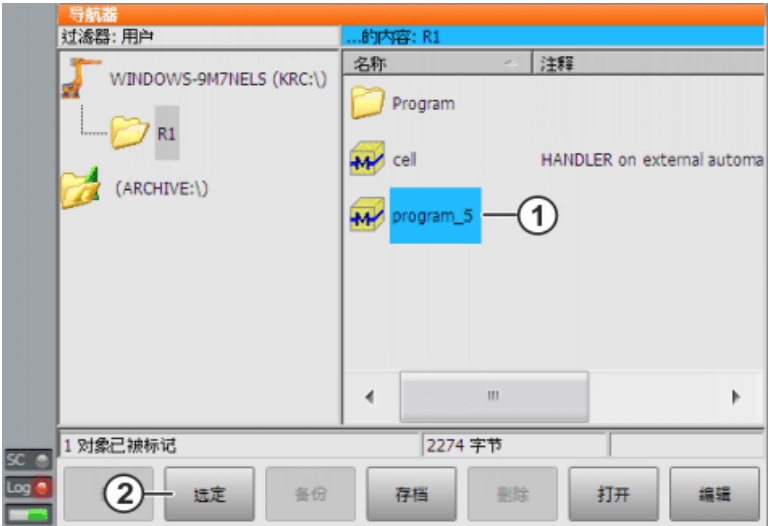


图 5-7：程序选择

2. 设定程序速度（程序倍率，POV）



图 5-8：POV 设置

3. 按使能键



图 5-9：使能开关

4. 按下启动键 (+) 并按住：
- 运行“INI”行，将模块所需的坐标系和工艺程序包初始化。

• 机器人执行 BCO 运行。



图 5-10：程序执行方向：向前/向后

序号	说明
1	启动正向运行按键
2	启动反向运行按键



警告

如果选定的运动语句包括 PTP 运行指令，则 BCO 运行将作为 PTP 运动从实际位置移动到目标位置。如果选定的运动语句包括 LIN 或 CIRC，则 BCO 运行将作为 LIN 运动被执行。必须观察运动过程，以避免碰撞。在 BCO 运行中速度自动降低。

5. 在达到目标位置之后停止运动



图 5-11

显示提示信息“达到了 BCO”。

6. 其他流程（根据设定的运行方式）：



图 5-12：激活驱动装置

- T1 和 T2：通过按启动键继续执行程序。

• AUT：激活驱动装置。
- 通过移动条件窗口中的驱动装置按钮激活驱动装置。
- 接着按动 Start （启动）启动程序。

- 在 Cell 程序中将运行方式转调为 **EXT** 并由 PLC 传送运行指令。

重置和取消选择程序

点击程序状态时，将打开一个弹出菜单，可用于取消选择或重置一个已选定的程序。

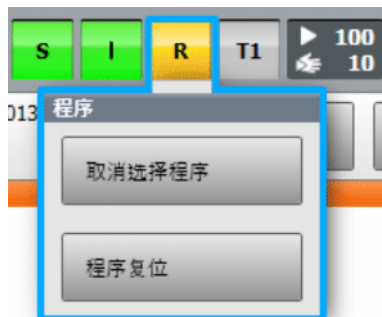


图 5-13: 重置程序，取消选择

5.5.1 练习：执行机器人程序

练习内容

打开课程练习本并完成以下练习：

- 执行机器人程序

6 程序文件的使用

6.1 学习单元：使用程序文件

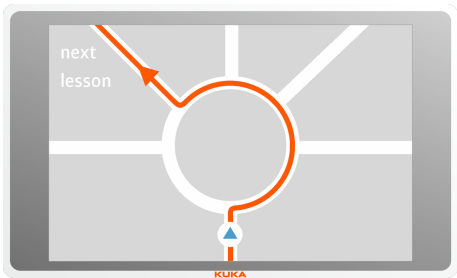


图 6-1：导航器

本学习单元讲授以下内容：

- 创建和编辑程序模块
- 存档和还原机器人程序
- 通过运行日志查看程序修改记录

6.2 创建程序模块

导航器中的程序模块

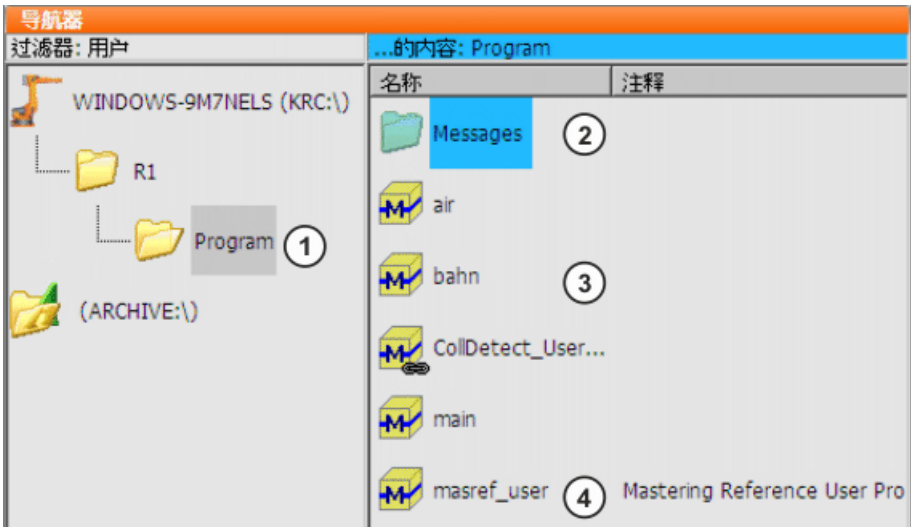


图 6-2：导航器中的模块

- 1 程序的主文件夹：“Program”
- 2 子文件夹
- 3 程序模块/模块
- 4 程序模块的注释

- 程序模块应保存在文件夹“Program”（程序）中。
- 也可建立新的文件夹并将程序模块存放在那里。
- 模块用字母“M”标示。
- 模块中可以加入注释（例如简要的功能说明）

程序模块的属性

模块由两个部分组成：

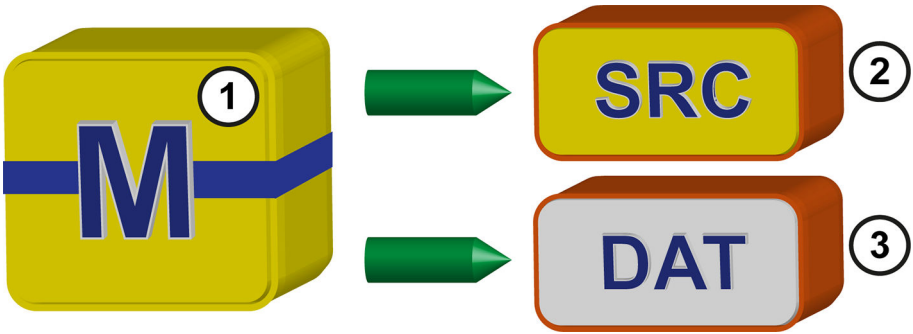


图 6-3：用户和专家视图中的模块/程序或子程序

- 1 用户界面中的模块视图
- 2 源文件：含有程序指令
在选择模块时在用户层面上显示
- 3 DAT 文件：含有已保存的位置



通过权限管理，*.SRC 和 *.DAT 文件默认只能在专家用户组中编辑。用户通常只能以只读模式显示视图。

- **源代码：**
SRC 文件中含有程序源代码。

```
DEF MAINPROGRAM()  
INI  
SPTP HOME Vel= 100% DEFAULT  
SPTP P1 Vel=100% PDAT1 TOOL[1] BASE[2]  
SPTP P2 Vel=100% PDAT2 TOOL[1] BASE[2]  
...  
END
```

- **数据列表：**
DAT 文件中含有固定数据和点坐标。

```
DEFDAT MAINPROGRAM()  
DECL E6POS XP1={X 900, Y 0, Z 800, A 0, B 0, C 0, S 6, T 27, E1 0, E2 0, E3 0, E4 0, E5 0, E6 0}  
DECL FDAT FPOINT1 ...  
...  
ENDDAT
```

创建程序模块

- 1. 在目录结构中选中要在其中创建程序的文件夹（1），例如程序文件夹。

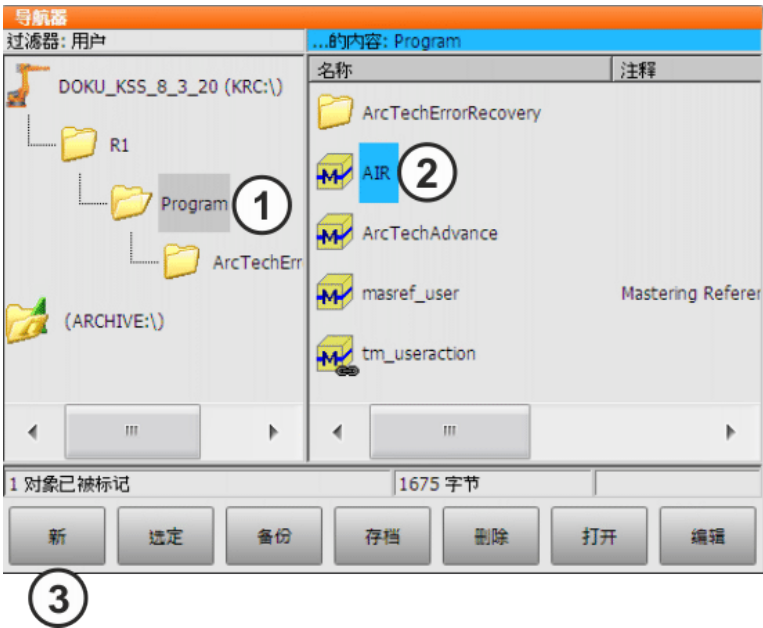


图 6-4：创建模块

如果要在该文件内创建一个模块，则应将焦点调整到文件夹（2）中。

2. 借助**新建**软键（3）创建一个新的模块
3. 指定程序名称

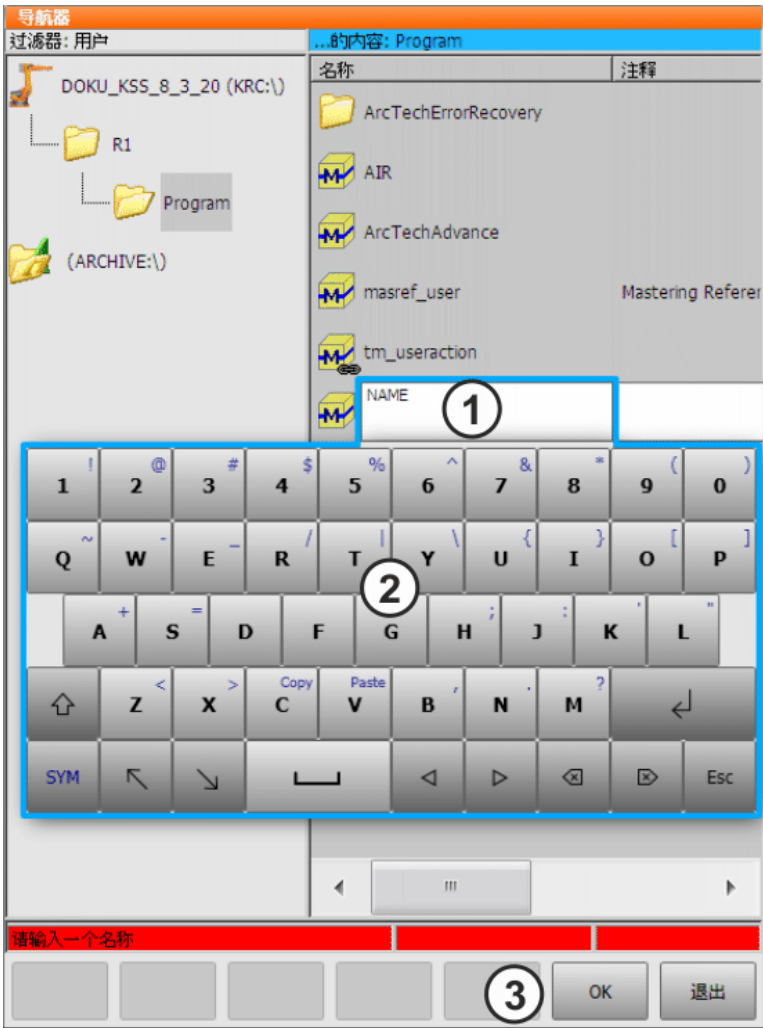


图 6-5：程序名称

名称输入栏自动激活 (1)。利用显示的键盘 (2) 进行输入。用 **OK** 键 (3) 确认输入内容。

6.3 编辑程序模块

编辑方式

与常见的文件系统类似，也可以在 KUKA smartPAD 导航器中编辑程序模块。
编辑方式包括：

- 复制
- 删除
- 重命名

操作步骤

复制程序

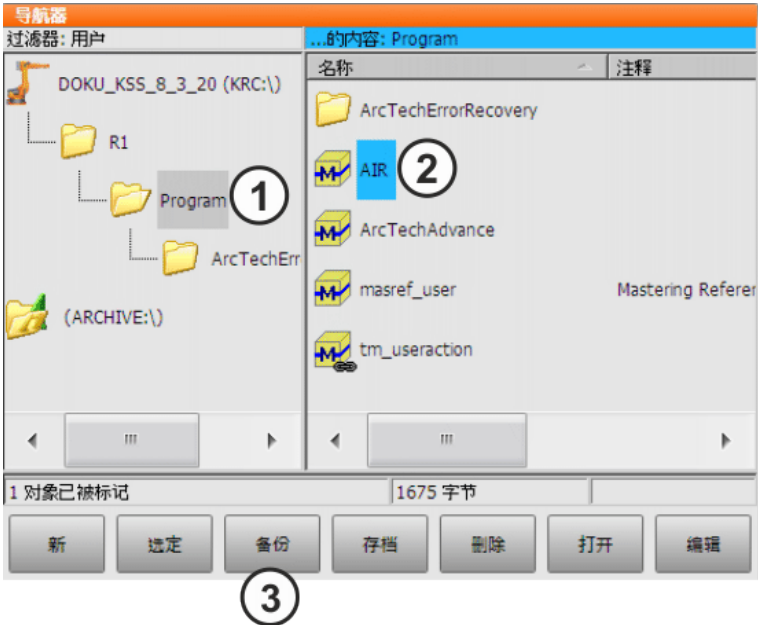


图 6-6：复制模块

1. 在文件夹栏中选中文件所在的文件夹 (1)。
2. 在文件列表中标记文件 (2)。
3. 选择**复制** (3) 按键。
4. 为新的模块分配一个新的程序名称。
5. 用 **OK** 键应用模块。



- 在用户组“专家”和筛选设置“详细信息”中，每个模块各有两个文件映射在导航器中（SRC 和 DAT 文件）。
- 必须复制这两个文件！

删除程序

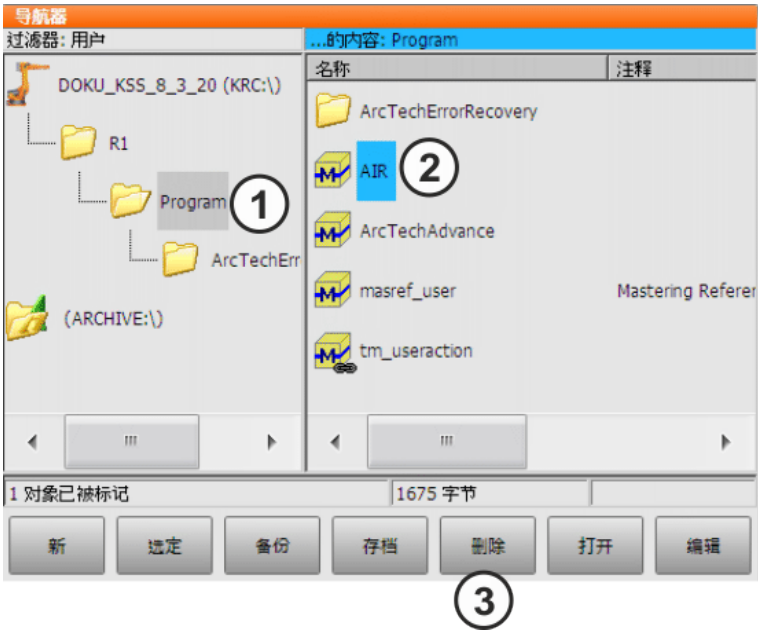


图 6-7：删除模块

1. 在文件夹栏中选中文件所在的文件夹（1）。
2. 在文件列表中标记文件（2）。
3. 选择软键**删除**（3）。
4. 用**是**确认安全询问。模块即被删除。



- 在权限管理的默认设置中，针对“专家”用户组和“详细信息”筛选设置，每个模块各有两个文件映射在导航器中（SRC 和 DAT 文件）。
- 必须删除这两个文件！
- 已删除的文件无法恢复！

重命名程序

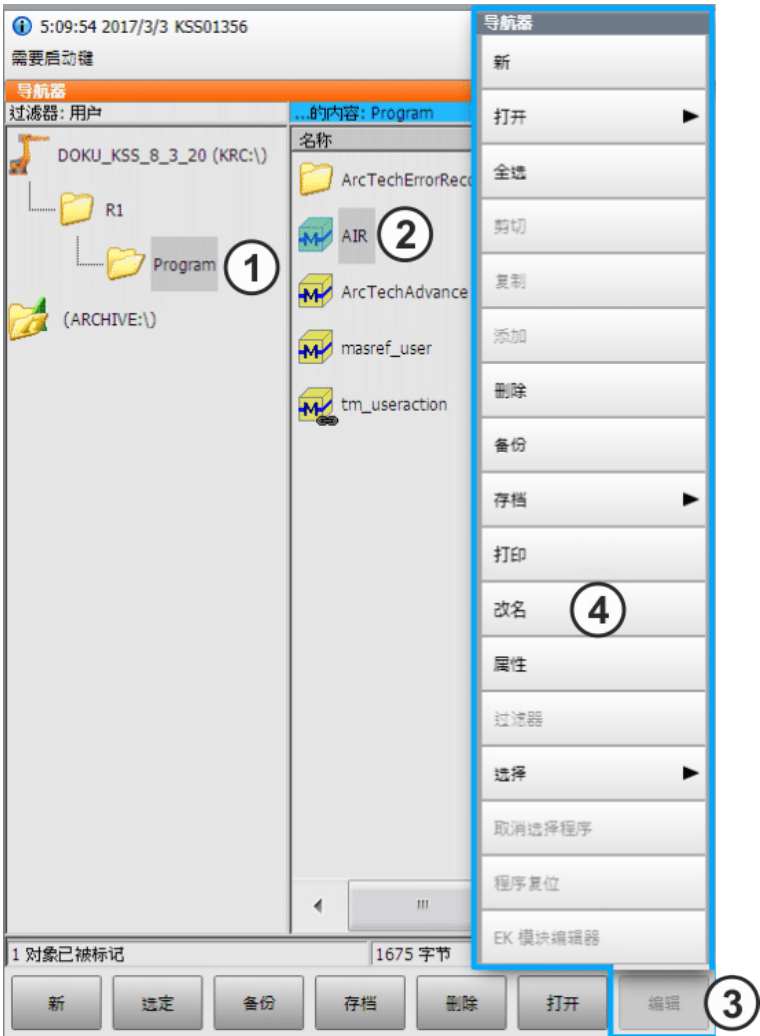


图 6-8：更名模块？

1. 在文件夹栏中选中文件（1）所在的文件夹。
2. 在文件列表中标记文件（2）。
3. 选择**编辑**（3）> **重命名**（4）按钮。
4. 用新的名称覆盖原文件名，并用 **OK** 确认。



- 在用户组“专家”和筛选设置“详细信息”中，每个模块各有两个文件映射在导航器中（SRC 和 DAT 文件）。
- 必须给这两个文件改名！

(>>> [12.3 “调整浏览器中的视图” 页面 337](#))

6.4 存档和恢复机器人程序

说明

已删除的文件无法恢复！

在存档过程中均会在相应的目标媒质上生成一个 ZIP 文件，该文件与机器人同名。该机器人名称可以在**机器人数据**菜单项中单独调整。

存储位置:



图 6-9: KUKA U 盘

有三个不同的 USB 存储位置可供使用:

- USB (KCP)
 - KCP (smartPAD) 上的 U 盘
- USB (控制柜)
 - 机器人控制柜上的 U 盘
- 网络
 - 在一个网络路径上存档
 - 所需的网络路径必须在**机器人数据**选项下配置。



在存档过程中，除了将生成的 ZIP 文件保存在所选的存储媒质上之外，同时还在驱动器 D:\ 上储存另一个存档文件 (BackupAll.ZIP)。

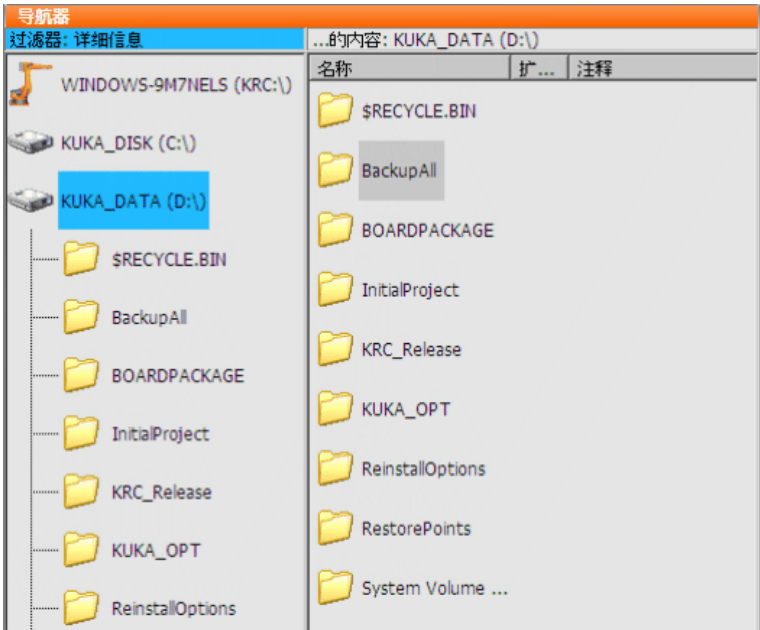


图 6-10: 存档

提示

第三方制造商的 U 盘可能导致数据丢失
在机器人控制器上操作时如果使用 KUKA 以外其他制造商的 U 盘，可能导致数据丢失。

- 对于机器人控制器上需要 U 盘的操作，请使用 KUKA U 盘。
KUKA U 盘已获得验证，可在机器人控制器上使用。

可选择以下数据存档:

- **所有:**
将恢复当前系统所需的数据存档。
- **应用:**
所有用户自定义的 KRL 模块 (程序) 和相应的系统文件均被存档。

- **系统数据:**
将设备数据存档。
- **日志数据:**
将 LOG 文件存档。
- **KRCDiag:**
将数据存档，以便将其提供给 KUKA 热线进行故障分析。生成一个 ZIP 文档，可通过电子邮件便捷发送。该文档还位于 C:\KUKA\KRCDiag 下。

操作步骤

数据存档

1. 将 U 盘插到选择的 USB 接口中。



图 6-11: KUKA U 盘

2. **菜单路径:** 机器人按键 > 文件 > 存档 > USB (控制柜) > 所需的子项

主菜单		文件	存档
文件	▶	存档	▶ USB (KCP) ▶
配置	▶	还原	▶ USB (控制柜) ▶
显示	▶	备份管理器	▶ 网络 ▶
诊断	▶		运行日志

图 6-12: 存档，菜单路径

3. 选择要存档哪些内容。

USB (控制柜)	
所有	
应用	
系统数据	
Log 数据	
KrcDiag	

图 6-13: 选择

4. 用**是**确认安全询问。
当存档过程结束时，信息窗口将显示一条信息。
5. 当 U 盘上的 LED 指示灯熄灭之后，可将其取下。



优先通过 USB (控制柜) 这种方式进行存档。此时 KCP-KUKA PC 可以直接通过主板端口写入 U 盘。相反，如果使用 smartPAD 上的接口，KPC 的数据必须通过现场总线系统 EtherCAT 传输到 smartPAD。这样绕弯路耗时长得多。

恢复数据



警告

通常情况下，只允许载入具有相应软件版本的存档。如果载入其他存档，则可能出现以下后果：

- 故障信息
- 机器人控制系统无法运行
- 人员受伤及财产损失

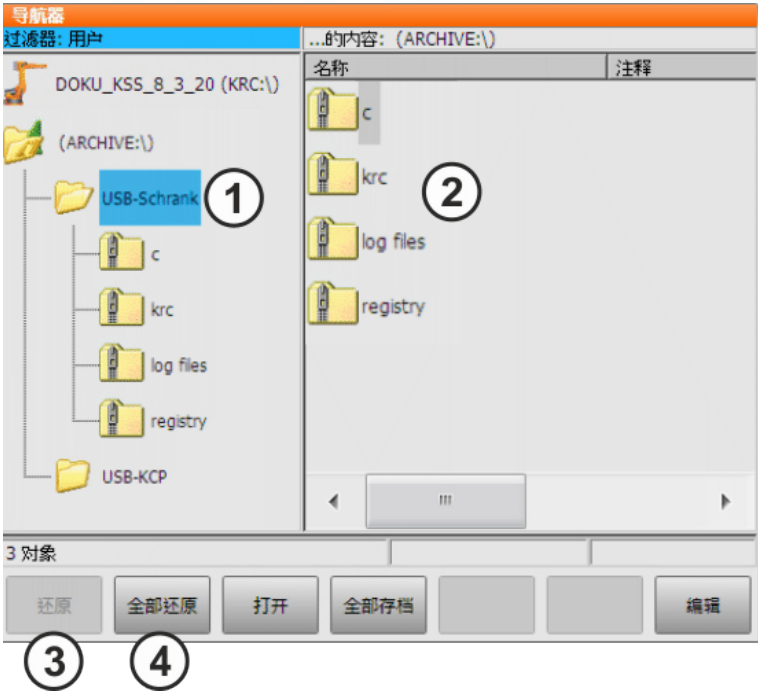


图 6-14：恢复存档

1. 将带有备份的 U 盘插入控制系统。
也可以选择使用网络存档路径。
2. 在 smartPAD 上在导航器中打开要恢复的存档（1）。
3. 也可以选中存档中的单个元素（2）。
4. 通过同名按钮**恢复**（3）单选内容。
以下菜单项可供选择：
 - 全部
 - 应用
 - 系统数据
5. 用**是**确认安全询问。
已存档的文件在机器人控制系统里重新恢复。
6. 如果要恢复所有内容，则使用按钮**全部恢复**（4）。
7. 只有当 U 盘上的 LED 熄灭之后方可拔出 U 盘。
否则会导致 U 盘受损。



系统会在以下情况时发出故障信息：

- 已存档文件版本与系统中的文件版本不同时。
- 应用程序包的版本与已安装的版本不一致时。

6.5 通过日志了解程序和状态变更

说明



图 6-15: 日志

- 控制系统上的所有操作、错误信息和状态都记录在日志中。
- 每个事件都有一个时间戳和一个序号。
- 日志不能删除。

操作步骤

1. 菜单路径: *KUKA 键* > *诊断* > *日志* > *显示*



图 6-16: 显示和配置日志

2. 日志将显示在 smartPAD 上。



图 6-17：运行日志，选项卡 Log

序号	说明
1	日志事件的类型 各个筛选类型和筛选等级均列在选项卡 筛选器 中。
2	日志事件的编号
3	日志事件的日期和时间
4	日志事件的简要说明
5	所选日志事件的详细说明
6	显示有效的筛选器

日志事件的筛选

- 在筛选器选项卡中可以设置显示日志中的哪些条目。
- 为此借助鼠标选中相关条目。

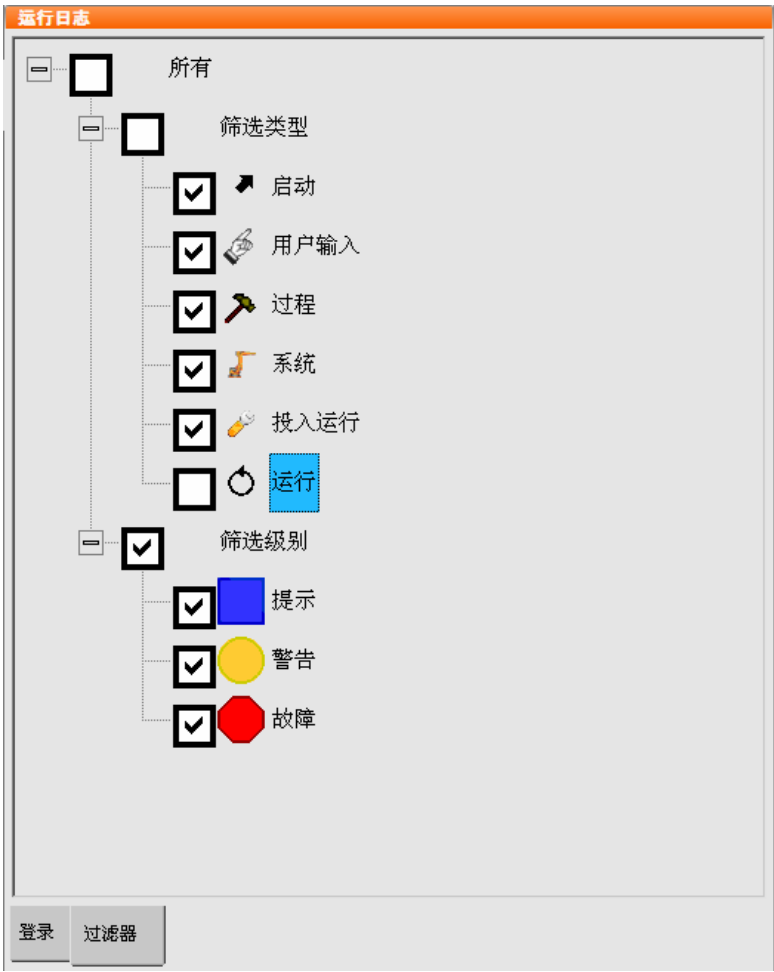


图 6-18：运行日志，选项卡筛选器

6.5.1 导出日志

说明

- 可以将当前版本的日志提取到文本文件中。
- 导出功能对控制系统的原始日志内容没有影响。

操作步骤

1. 至少需要**应用人员**用户组。
2. 在默认设置中，导出功能至少需要**专家**用户组。
3. **菜单路径：**诊断 > 日志 > 配置
4. 配置导出参数。



图 6-19：窗口配置运行日志

1	应用输出的筛选设置。如果不勾选，则在输出时不会进行筛选。
2	文本文件路径。
3	因缓冲溢出而删除的日志数据会以灰色背景显示在文本文件中。

- 5. 按下 **OK** 按钮保存配置并关闭窗口。
- 6. 切换到日志显示
菜单路径：*KUKA 键* > *诊断* > *日志* > *显示*
- 7. 通过同名按钮（1）**导出**日志



图 6-20：导出日志

7 建立及更改编程的运动

7.1 学习单元：创建和更改编程的运动

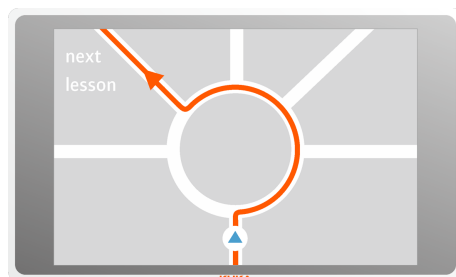


图 7-1：导航器

本学习单元讲授以下内容：

- 创建新的运动指令
- 创建最优时间运动
- 创建连续轨迹运动（CP 运动）
- 用全局点编程
- 更改运动指令

7.2 创建新的运动指令需要哪些信息？

机器人运动编程

如果必须对机器人运动进行编程，需要解决以下诸多问题：

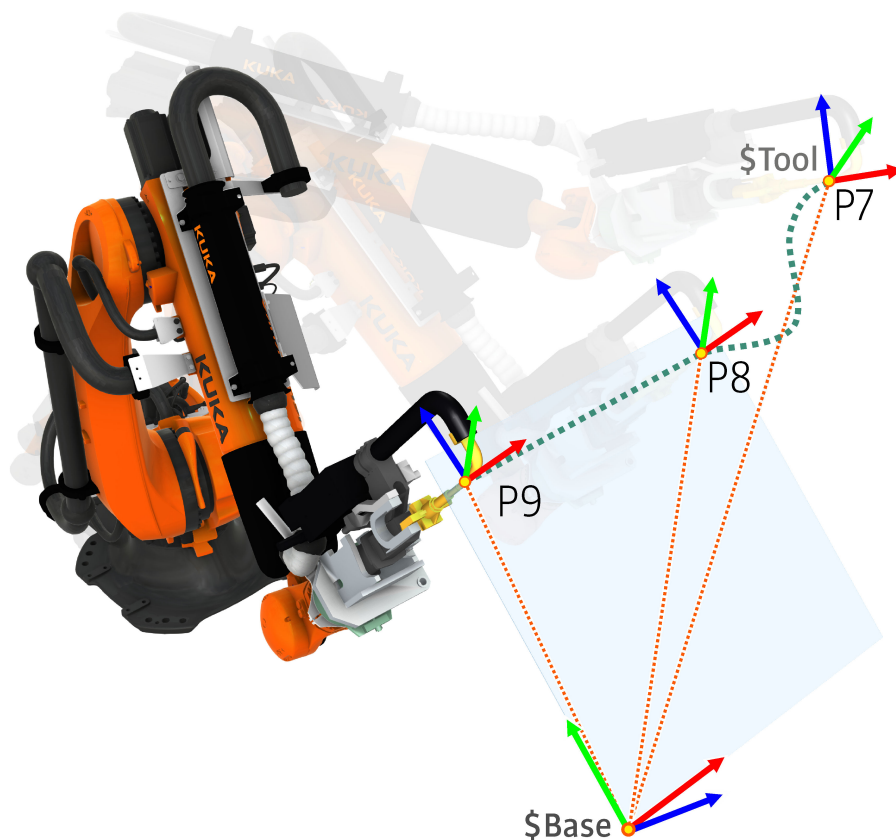


图 7-2：机器人运动

问题	解决方案	关键词
机器人如何记住其位置？	工具在空间中的相应位置会被保存。（机器人位置对应于所设定的工具坐标和基坐标）	POS E6POS
机器人如何知道它必须如何运动？	通过指定运动方式：点到点，直线或者圆形。	SPTP SLIN SCIRC
机器人运动的速度有多快？	两点之间的速度和加速度可通过编程设定。	Vel. Acc.
机器人是否必须在每个点上都要停住？	为了缩短节拍时间，点也可以轨迹逼近，但这样就不会精确暂停。	CONT
如要到达某个点，工具会沿哪个姿态？	可以针对每个运动对姿态引导进行单独设置。 该设置仅对轨迹运动有效。（>>> “运动方式” 页面 186）	ORI_TYPE
机器人是否会识别障碍？	不会，机器人只会“坚定不移”地沿编程设定的轨迹运动。编程人员要负责保证移动时不会发生碰撞。 但也有用于保护机器的“碰撞监控”方式。	碰撞监控

运动方式

- 有不同的运动方式供运动指令的编程使用。
- 可根据对机器人工作流程的要求来进行运动编程。
 - 单轴运动
SPTP » 点到点

- 轨迹运动:
 - SLIN » 直线
 - SCIRC » 圆周



PTP、LIN 和 CIRC 运动不包括在本培训资料之内。详细信息可查阅 [Xpert](#) 中 *KUKA System Software 8.x* 的操作及编程指南。

借助行指令编程

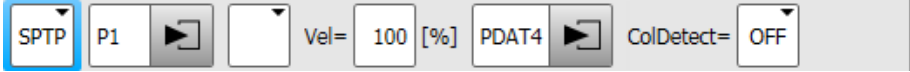


图 7-3：行指令

- 用示教方式对机器人运动进行编程时必须传输上述的点和运动特定的信息。
- 为此应使用可以很方便地输入这些信息的行指令。

7.3 SPTP - 最优时间运动（轴运动）

说明

SPTP - *Spline Point To Point* 是一种轴特定的运动。机器人将工具的 TCP 沿**最快速**轨迹运动到目标点。最快速的轨迹通常并不是最短的轨迹，即**不是**直线。由于机器人轴旋转运动，因此弧形轨迹会比直线轨迹更快。程序中的第一个运动必须为 SPTP 运动，因为只有在此运动中才会分析状态和转角方向（(>>> [7.3.1 “状态和转角方向 \(Status & Turn\)” 页面 189](#))). 示教点后，运动的具体过程不可预见。但是运行过一次后，机器人将不断以相同的方式重复运动。

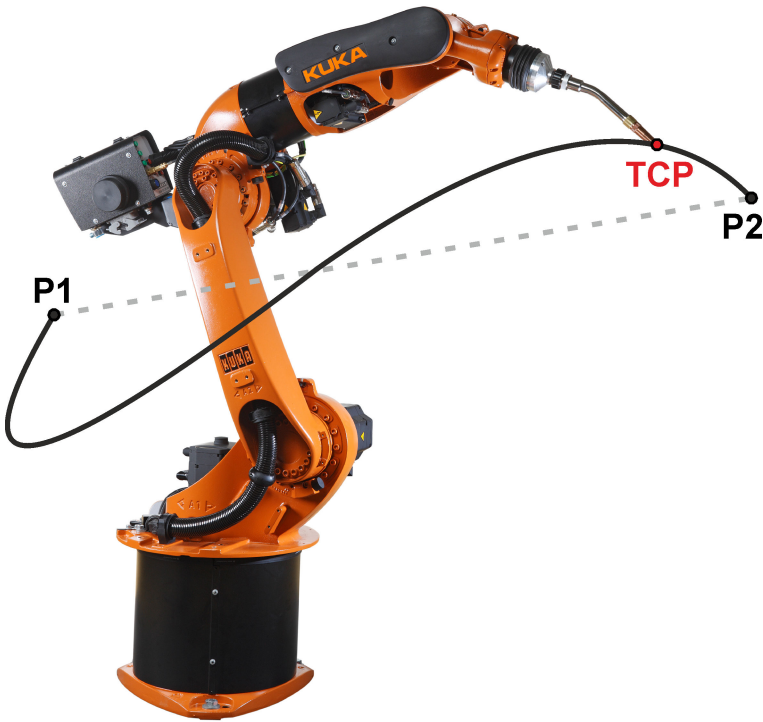


图 7-4: SPTP 运动

同步 PTP（点对点）

在同步 PTP 时所有轴共同启动并同时停下。其中，行指令中给出的速度也考虑在内。主导轴是到达目标点所需时间**最长**的轴。

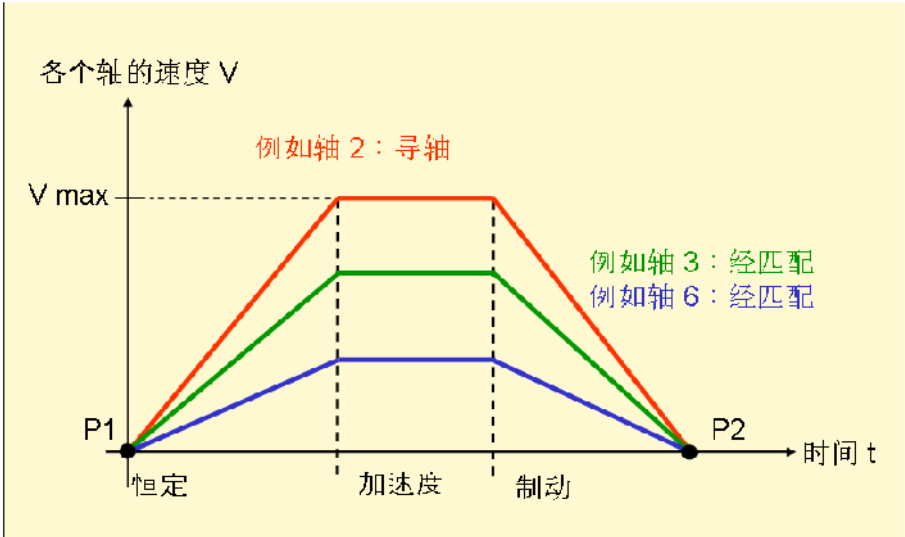


图 7-5: 同步 PTP（点对点）

使用

机器人要尽量快速并有针对性地接近点时，需使用 SPTP 运动。此时轨迹处于次要位置。为确保运动不会发生碰撞，可通过轨迹逼近的中间点将机器人

“引导”至目标。也可以借助轨迹逼近的 SPTP 运动，优化至轨迹运动或轨迹运动之间的机器人运行的节拍时间。

示例

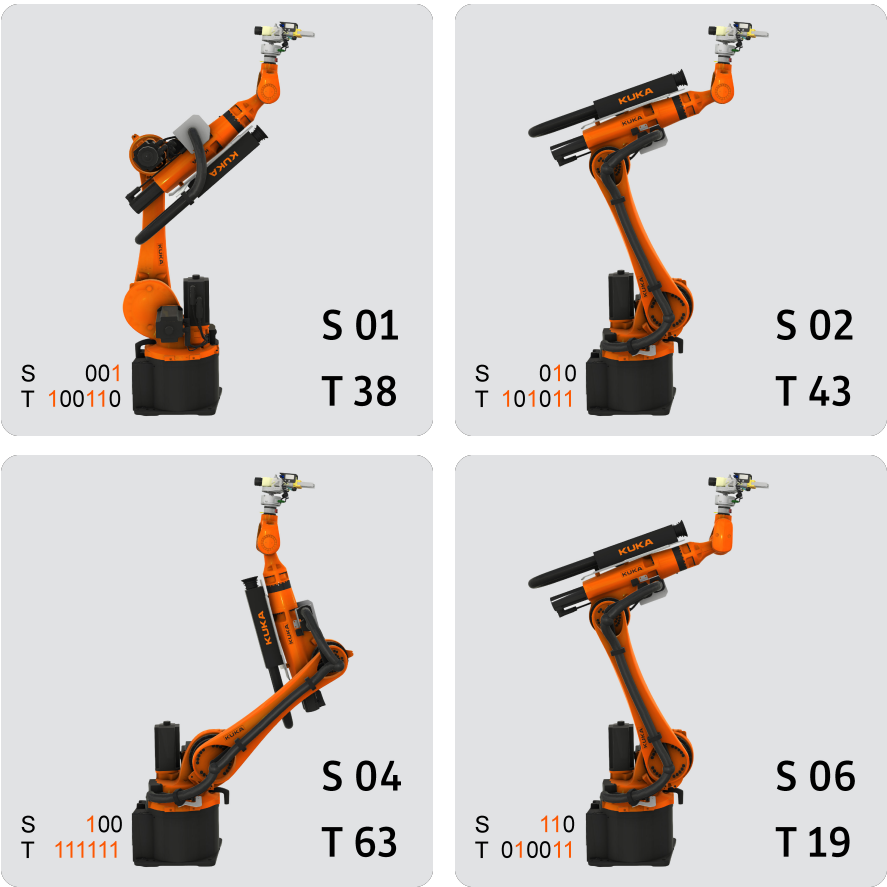
- 点焊和铆接
- 运输
- 测量，检验

7.3.1 状态和转角方向 (Status & Turn)

说明

状态和转角方向的作用

状态和转角方向用于从 TCP 同一位置的多个可能的轴位中确定唯一的轴位。由于状态值和转角值的不同，轴位置也随之不同：



状态和转角方向分析

机器人控制器仅在 SPTP 运动时才会考虑编程设置的状态和转角方向值。在轨迹运动 (SLIN、SCIRC) 和样条组 (SPL) 时会将其忽略。程序中的第一个运动指令是 SPTP (HOME)。

- HOME 位置的类型是 E6AXIS (config.dat)，它定义了唯一的起始位置。
- SPTP 其他可能的类型有 AXIS、E6POS 和 POS

```
DEFDAT MAINPROGRAM ()  
DECL POS XPOINT1={X 900, Y 0, Z 800, A 0, B 0, C 0, S 6, T 27}  
DECL FDAT FPOINT1 ...  
...
```

7.3.2 SPTP 轨迹逼近

轨迹逼近

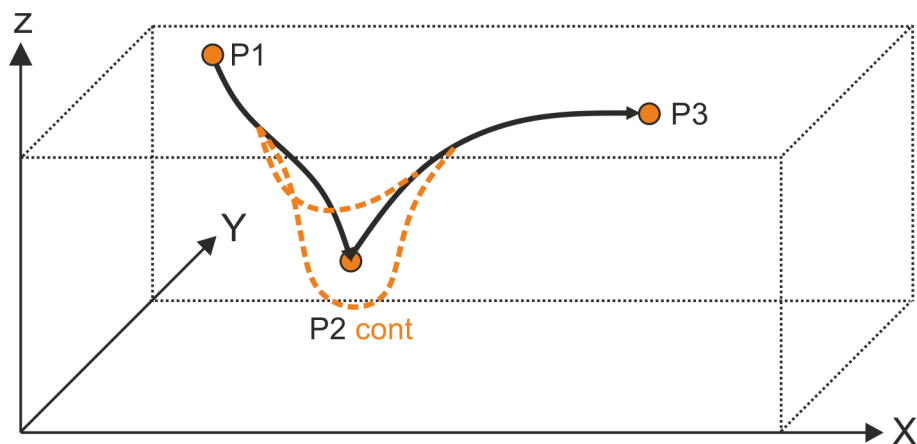


图 7-6: (S)PTP 点的点轨迹逼近

- 为了加速运动过程，控制器可以 CONT 标示的运动指令进行轨迹逼近。
- 轨迹逼近意味着将不精确移到点坐标。
- 事先便离开精确保持轮廓的轨迹。
- TCP 被导引沿着轨迹逼近轮廓运行，该轮廓止于下一个运动指令的精确保持轮廓。

轨迹逼近的优点

机器人在各点之间无需再制动和加速 (1)。由此可以更快地执行程序 (2)，从而优化节拍时间，提高能效。

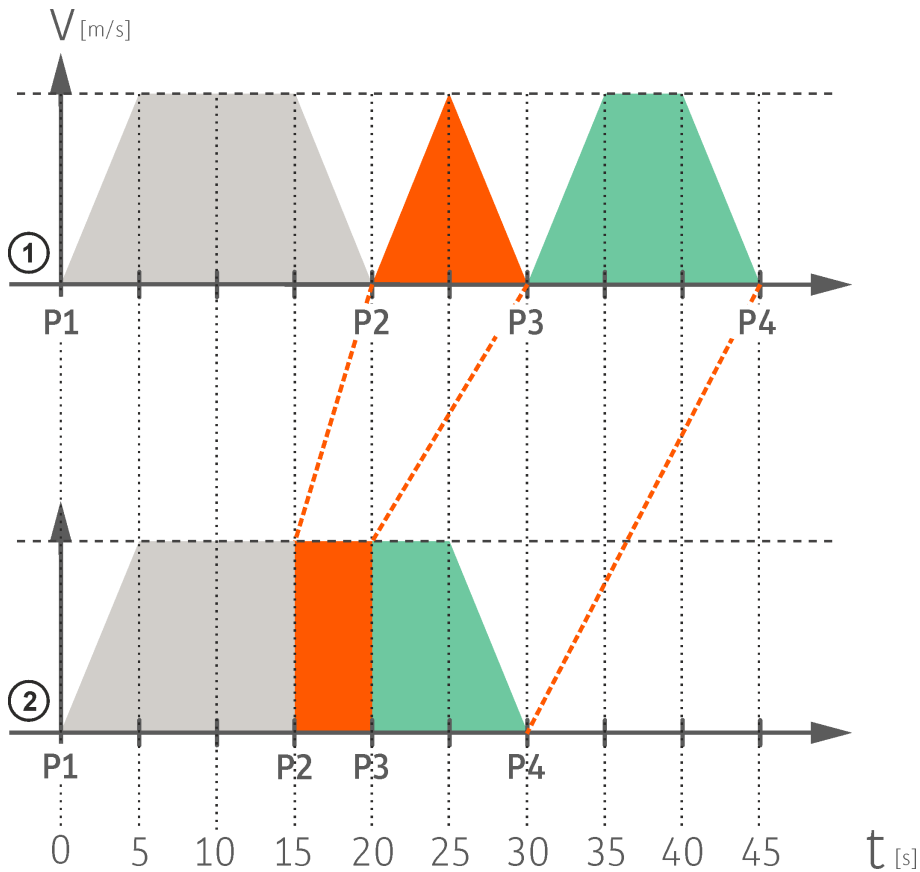


图 7-7：比较精确暂停和轨迹逼近

轨迹逼近时的 SPTP 轨迹特性

为了能够执行轨迹逼近运动，控制器必须能够载入后续运动指令。通过计算机预进读入。轨迹逼近距离的单位为 mm。

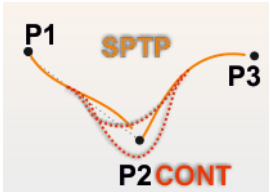


图 7-8



轨迹逼近不可预见！

7.3.3 借助行指令对运动编程

前提条件

- 已设置运行方式 T1
- 已选定机器人程序

操作步骤

1. 将 TCP 移向应被设为目标点的位置。

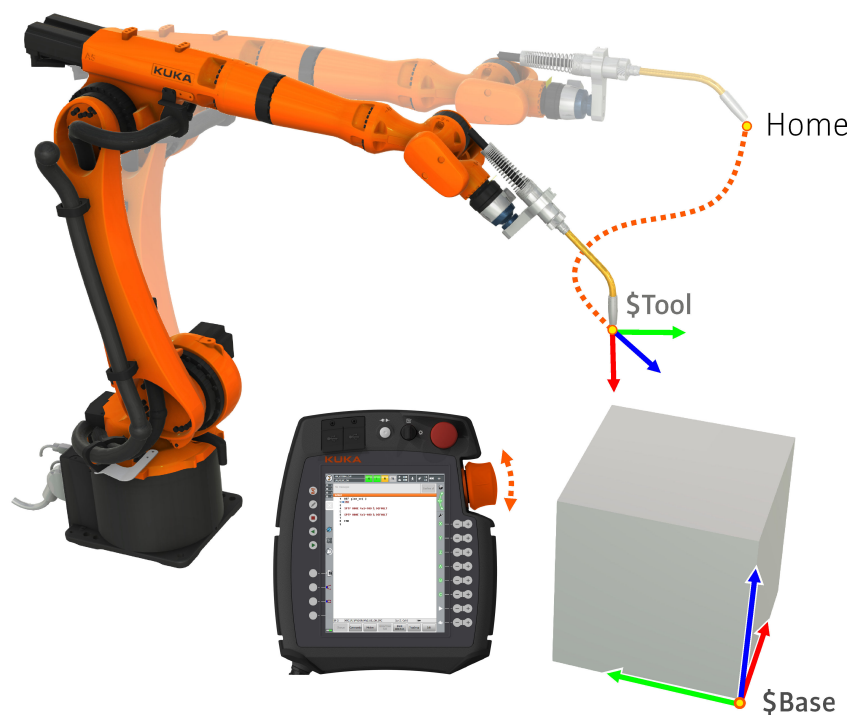


图 7-9: 运动指令

2. 插入行指令

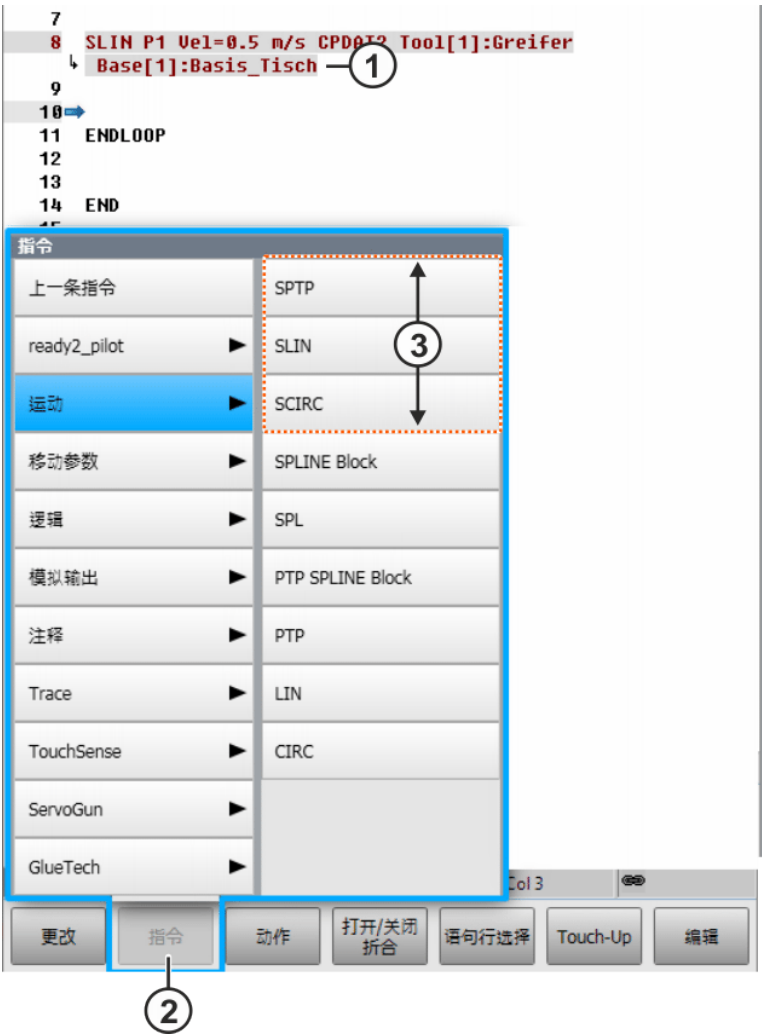


图 7-10：插入行指令

- 光标选中要在其后面插入运动指令的那一行 (1)。
- 菜单序列 **指令** (2) > **运动** > **SPTP**

另一种方法：



图 7-11：插入运动

- 光标选中要在其后面插入运动指令的那一行 (1)。
 - 通过同名按钮插入一个 **运动** (2)。
3. 一个预填的行指令将插入在选中的程序行后面。



图 7-12：选择运动方式

如果通过运动按钮插入了行指令，则会自动打开运动选项（1）。SPTP（2）已预选。

- 4. 配置各个运动参数。
 - SPTP
(>>> 7.3.3.1 “行指令：SPTP” 页面 195)
- 5. 用指令 **OK** 保存指令。TCP 的位置被作为目标点示教。

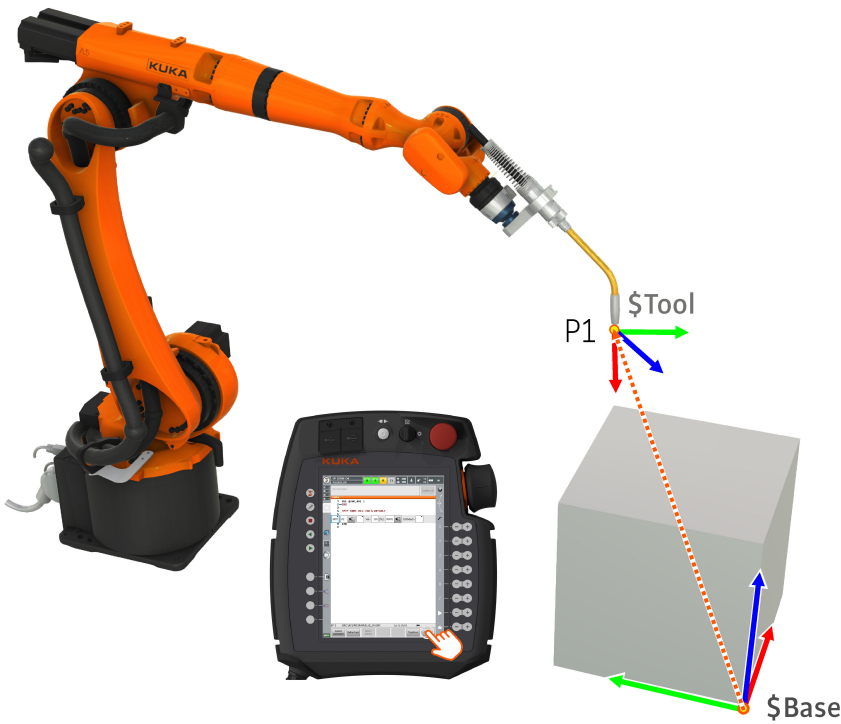


图 7-13：在“指令 OK”和“Touchup”时保存点坐标

7.3.3.1 行指令：SPTP

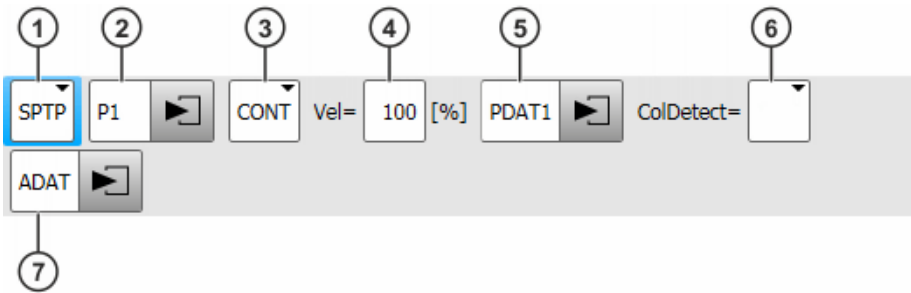


图 7-14：行指令 SPTP（单个运动）

序号	说明
1	运动方式 SPTP
2	目标点的名称。系统自动赋予一个名称。名称可以被改写。 箭头符号： (>>> “设置工具和基坐标” 页面 196)
3	- CONT：目标点被轨迹逼近。 [空]：将精确地移至目标点。
4	速度 1 ... 100 %
5	运动数据组名称。系统自动赋予一个名称。名称可以被改写。 箭头符号： (>>> “更改运动参数” 页面 196)
6	该运动的碰撞识别 - OFF：碰撞识别已关闭 - CDSets_Set[编号]：碰撞识别已开启。为了识别将使用数据组编号中的数值。
7	通过切换参数可显示和隐藏该栏目。 含逻辑参数的数据组名称。系统自动赋予一个名称。名称可以被改写。需要编辑数据时请触摸箭头。相关选项窗口即自动打开。

更改运动参数



图 7-15：运动参数

1. 在打开的行指令中点击 PDATx (1) 旁的箭头符号。
2. 在打开的弹出菜单“运动参数”中设置下列值。
 - 轨迹逼近距离 (2)
(>>> 7.3.2 “SPTP 轨迹逼近” 页面 190)
 - 加速度 (3)

设置工具和基坐标

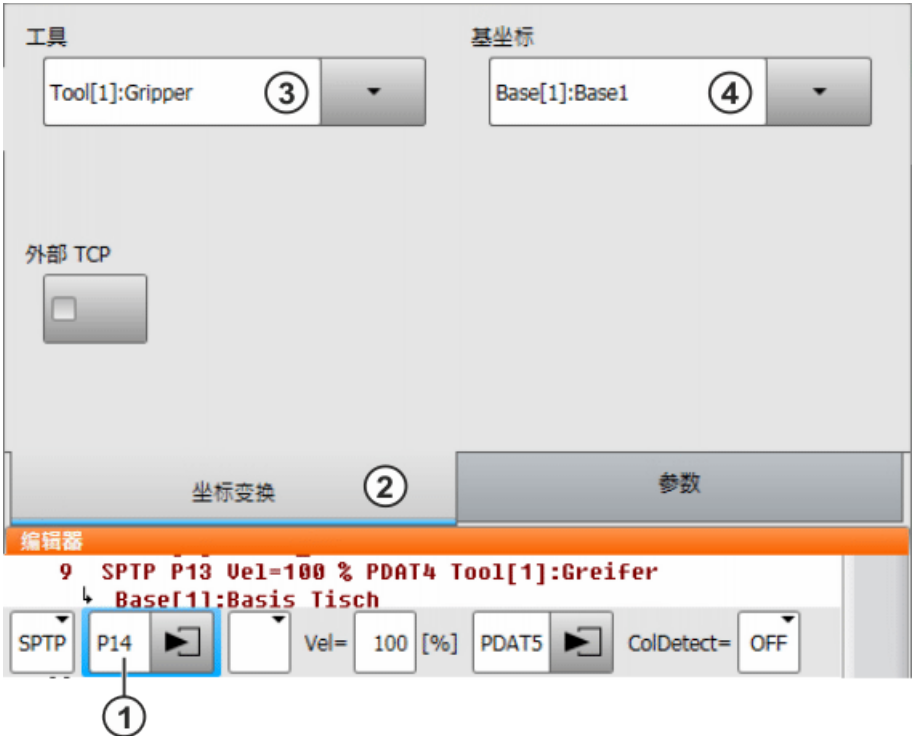


图 7-16：设置基坐标和工具

1. 在行指令中点击点名称旁边的箭头符号。
2. 在打开的弹出菜单中通过坐标系选项卡 (2) 设置工具 (3) 和基坐标 (4)

7.3.4 练习：空运转程序 - 程序操作和 SPTP 运动

练习内容

打开课程练习本并完成以下练习：

- 空运转程序 - 程序操作和 SPTP 运动

7.4 创建沿轨迹的运动

SLIN 和 SCIRC 运动方式

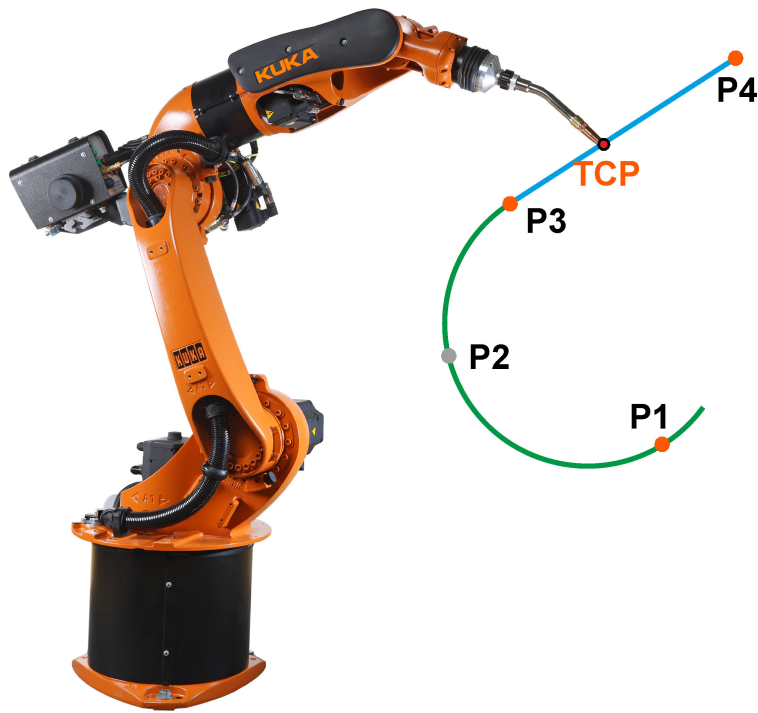
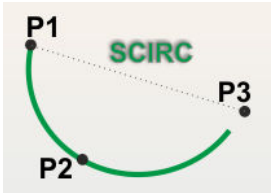


图 7-17：轨迹运动

运动方式	含义	应用示例
	<i>Linear</i>: 直线 - 直线型轨迹运动： - 工具的 TCP 按设定的姿态从起点匀速移动到目标点。 - 速度和姿态均以 TCP 为参照点。	轨迹应用，例如： - 轨迹焊接 - 粘接 - 激光焊接/切割
	<i>Circular</i>: 圆弧 - 圆形轨迹运动是通过起点、辅助点和目标点定义的 - 工具的 TCP 按设定的姿态从起点匀速移动到目标点。 - 速度和姿态均以工具的 TCP 为基准。	轨迹应用与 SLIN 相同： 圆周、半径、圆形

7.4.1 奇点

说明

有着 6 个自由度的 KUKA 机器人具有 3 个不同的奇点位置。
即便在给定状态和转角顺序的情况下，也无法通过逆向变换（将笛卡尔坐标转换成轴坐标值）得出唯一数值时，即可认为是一个奇点位置。这种情况下，或者当最小的笛卡尔变化也能导致非常大的轴角度变化时，即为奇点位置。奇点不是机械特性，而是数学特性，出于此原因，奇点只存在于 CP 运动范围内，而在轴运动时不存在。

过顶奇点

- 对于过顶奇点来说，腕点（即轴 A5 的中点）垂直于机器人的轴 A1。
- 轴 A1 的位置不能通过逆向运算进行明确确定，且因此可以赋以任意值。

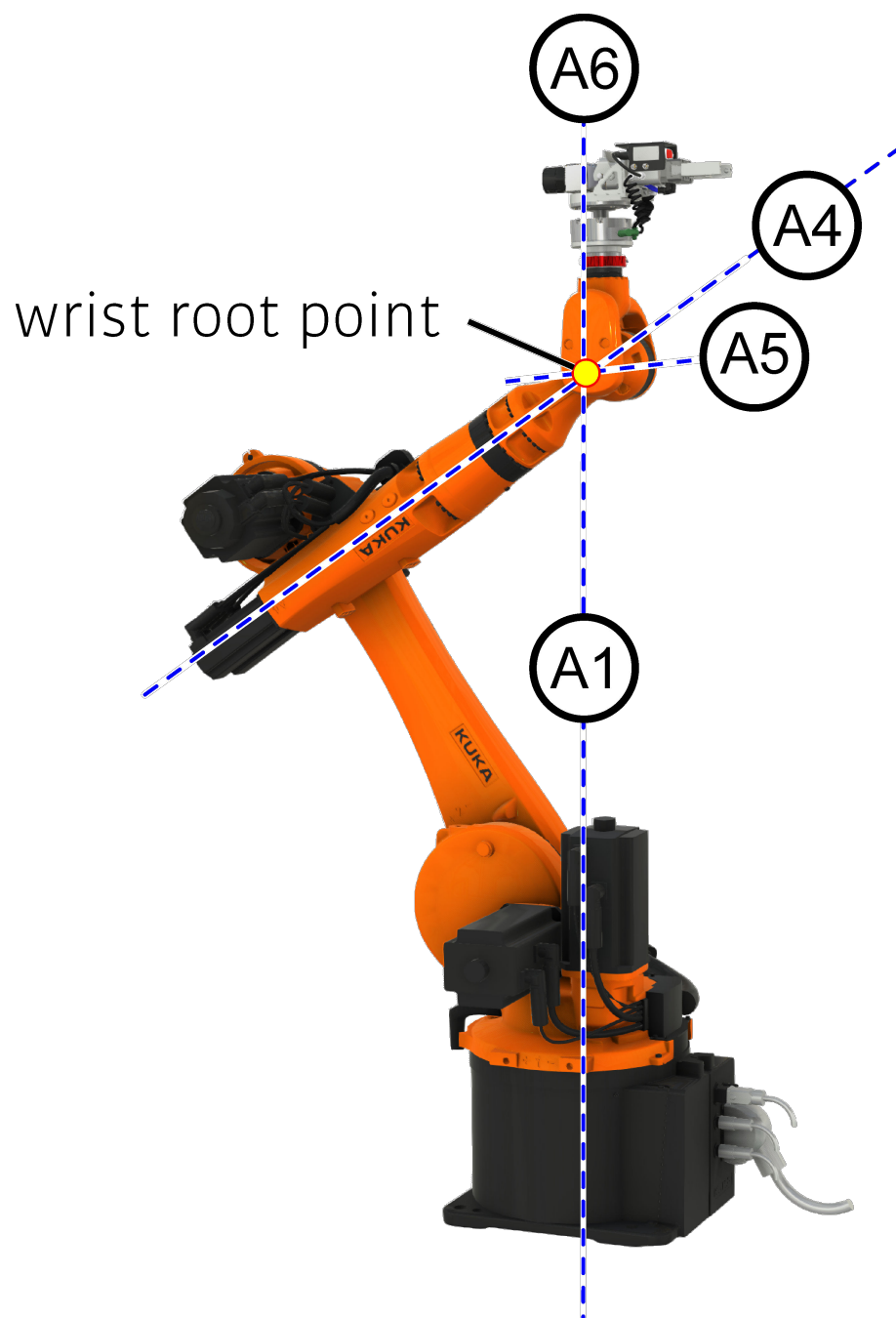


图 7-18：过顶奇点

延展位置奇点

- 对于延展位置奇点来说，腕点（即轴 A5 的中点）位于机器人的轴 A2 和 A3 的延长线上。
- 机器人处于其工作范围的边缘。
- 通过逆向运算将得出唯一的轴角度，但较小的笛卡尔速度变化将导致轴 A2 和 A3 较大轴速变化。

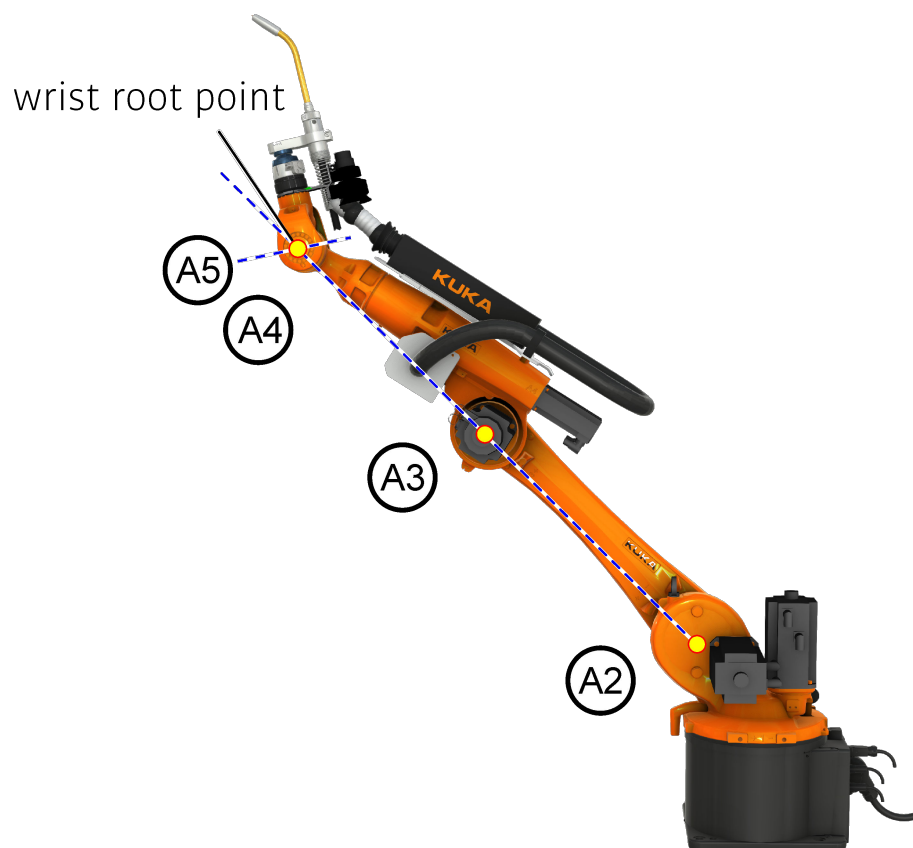


图 7-19: 延展位置

手腕轴奇点

- 对于手腕轴奇点来说，轴 A4 和 A6 彼此平行，并且轴 A5 处于 $\pm 0.01812^\circ$ 的范围内。
- 通过逆向变换无法明确确定两轴的位置。
- 轴 A4 和 A6 的位置可以有任意多的可能性，但其轴角度总和均相同。

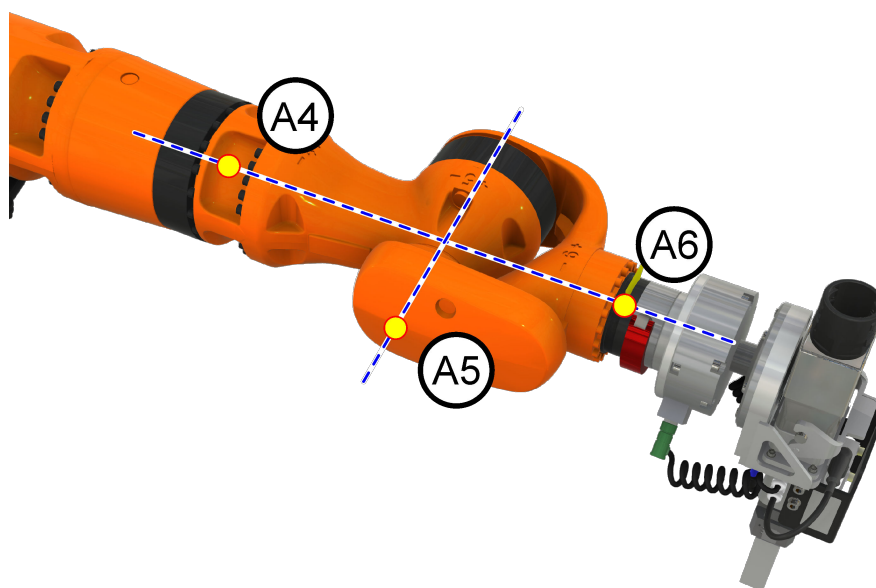


图 7-20: 手腕轴奇点

7.4.2 SLIN 编程

1. 将 TCP 移向应被设为目标点的位置。

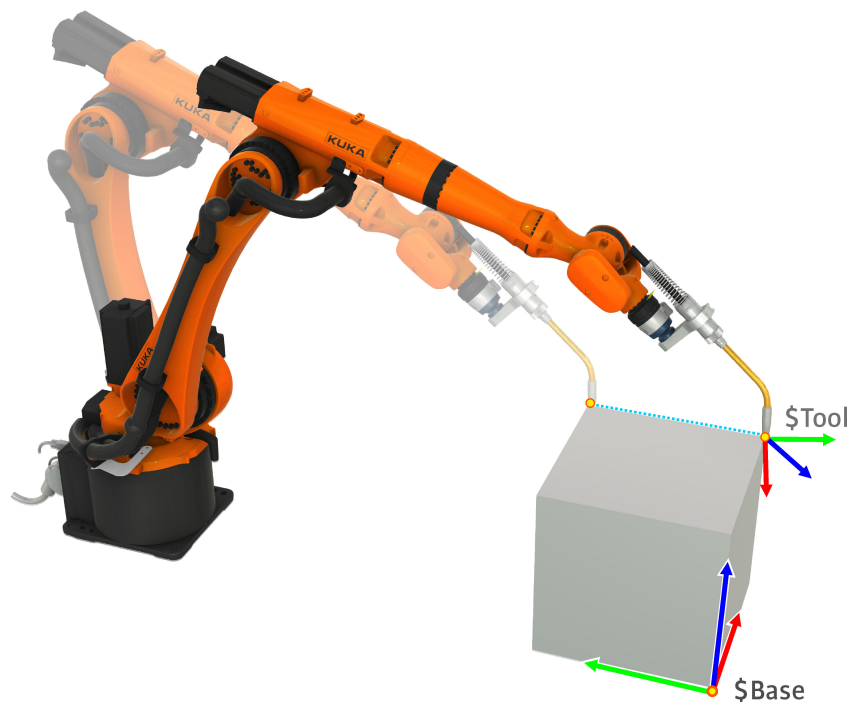


图 7-21：带 SLIN 的运动指令

2. 插入 SLIN 行指令

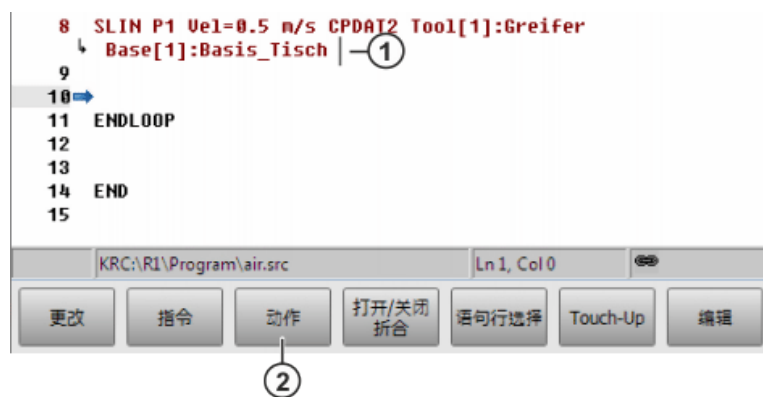


图 7-22：插入运动

- 光标选中要在其后面插入运动指令的那一行 (1)。
 - 通过同名按钮插入一个**运动** (2)。
3. 一个预填的行指令将插入在选中的程序行后面。

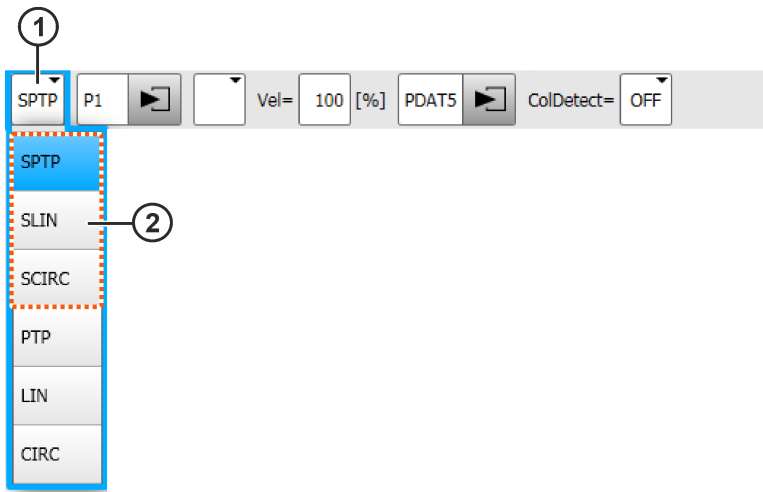


图 7-23：选择运动方式

- 在行指令中点击运动方式的栏 (1)。
 - 一个选择列表打开，在此选择运动方式 SLIN (2)。
4. 配置各个运动参数。
- SLIN
(>>> 7.4.2.1 “行指令：SLIN” 页面 204)
5. 用指令 OK 保存指令。TCP 的当前位置被作为目标点示教。

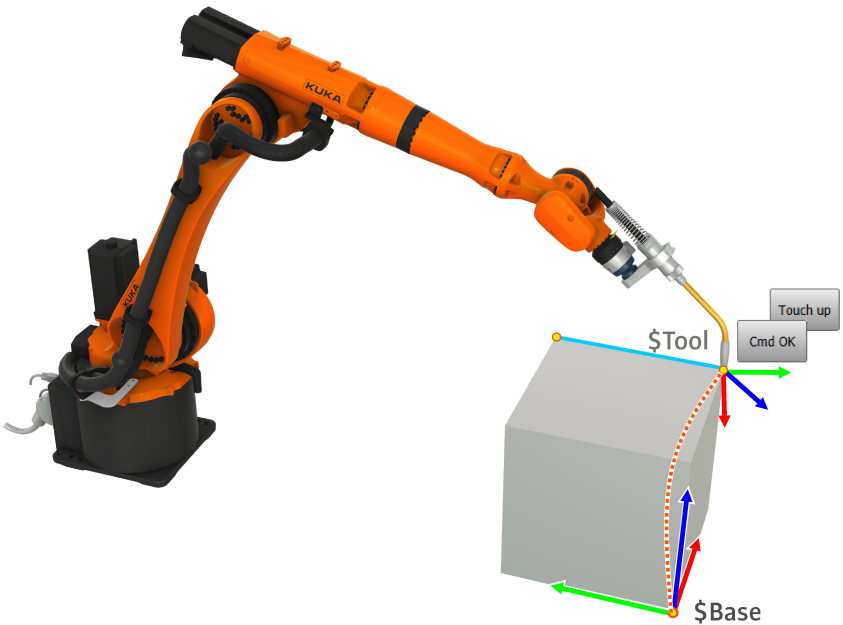


图 7-24：在“指令 OK”和“Touchup”时保存点坐标

7.4.2.1 行指令：SLIN

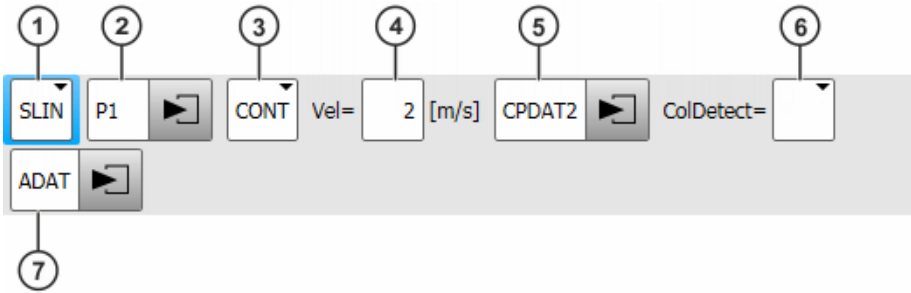


图 7-25: SLIN 行指令（单个运动）

序号	说明
1	运动方式 SLIN
2	目标点的名称。系统自动赋予一个名称。名称可以被改写。 箭头符号: (>>> “设置工具和基坐标” 页面 196)
3	• CONT: 目标点被轨迹逼近。 • [空]: 将精确地移至目标点。
4	速度 • 0.001 ... 2 m/s
5	运动数据组名称。系统自动赋予一个名称。名称可以被改写。 需要编辑点数据时请触摸箭头。相关选项窗口即自动打开。
6	该运动的碰撞识别 • OFF: 碰撞识别已关闭 • CDSets_Set[编号]: 碰撞识别已开启。为了识别将使用数据组编号中的数值。
7	通过切换参数可显示和隐藏该栏目。 含逻辑参数的数据组名称。系统自动赋予一个名称。名称可以被改写。需要编辑数据时请触摸箭头。相关选项窗口即自动打开。

姿态引导

- 轨迹运动时可以准确定义姿态引导。
- 工具在运动的起点和目标点处的方向可能不同。
 - 工具的姿态引导
(>>> [7.4.2.2 “SLIN 时的姿态引导” 页面 204](#))

7.4.2.2 SLIN 时的姿态引导

运动方式 SLIN

在运动方式 SLIN 下的姿态导引

- 标准或手动 PTP
工具的方向在运动过程中不断变化。
在机器人以标准方式到达手轴奇点之前就可以使用手动 PTP，因为是通过手轴角度线性转换（轴相关运行）进行姿态变化。

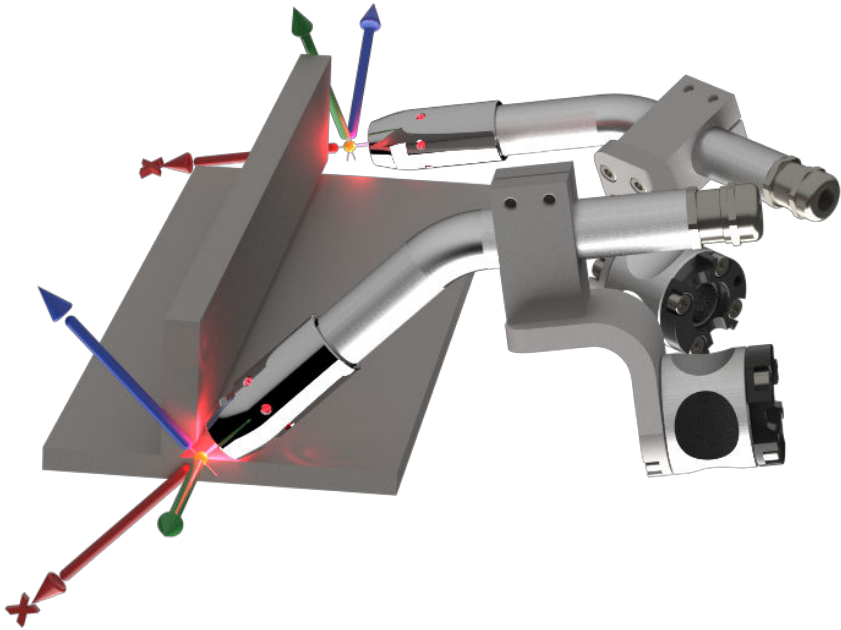


图 7-26：姿态引导：标准



小心

奇点可导致快速且显著的姿态变化。尤其是对于大型工具，偏转较大
会导致财产损失甚至人员受伤。

- **常数**
在起点示教的工具姿态在整个运动过程中保持不变（或得以维持）。在
终点示教的姿态被忽略（图中橙色工具的姿态）

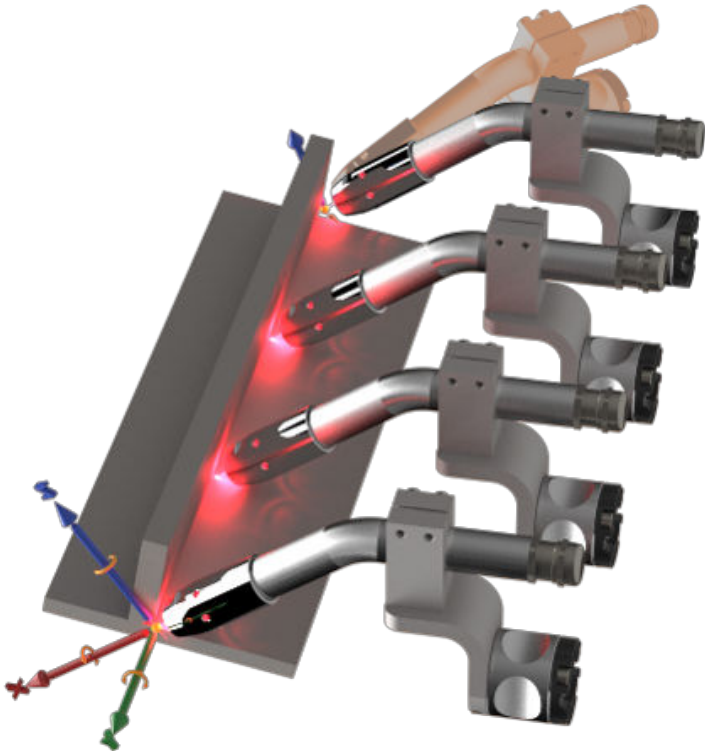


图 7-27：姿态引导：常数

操作步骤

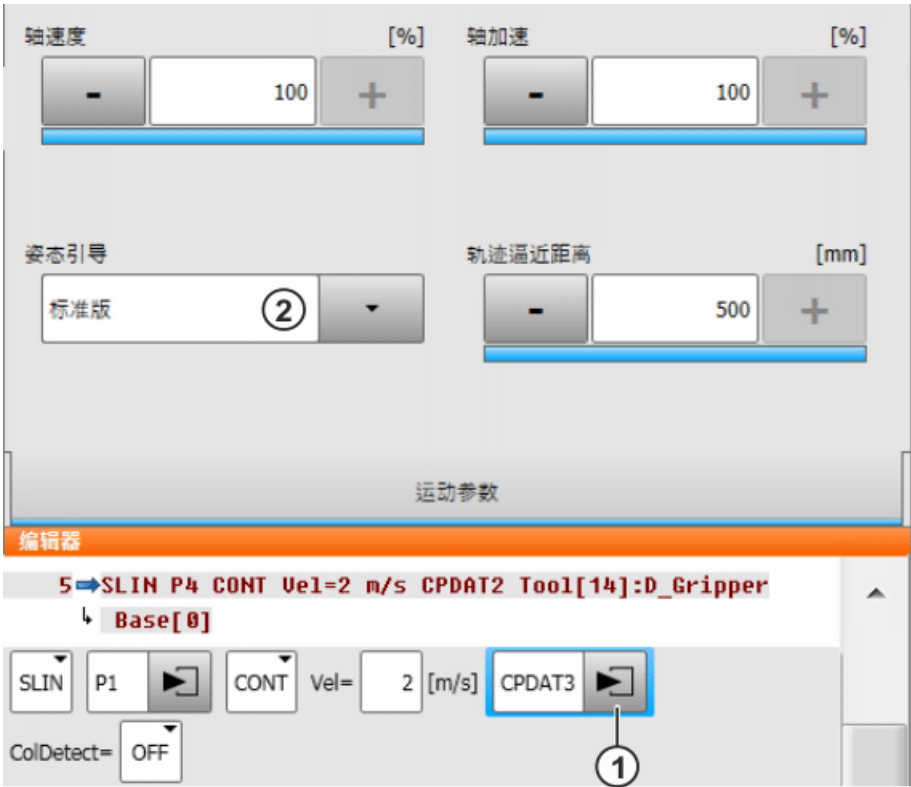


图 7-28: SLIN 运动参数

- 1. 在打开的行指令中点击 CPDATn 栏 (1) 的箭头符号。
- 2. 在显示的评估参数窗口中进行下列调整
 - 姿态引导 (2)
(>>> “运动方式 SLIN” 页面 204)
数值
 - 标准
 - 手动 PTP
 - 恒定的姿态引导

7.4.3 SCIRC 编程

进行 SCIRC 时的运动过程

在此，工具的参考点沿着圆弧运动到目标点。轨迹则由起始点、辅助点和目标点进行描述。前一个运动指令的目标点作为下一个指令的起始点。

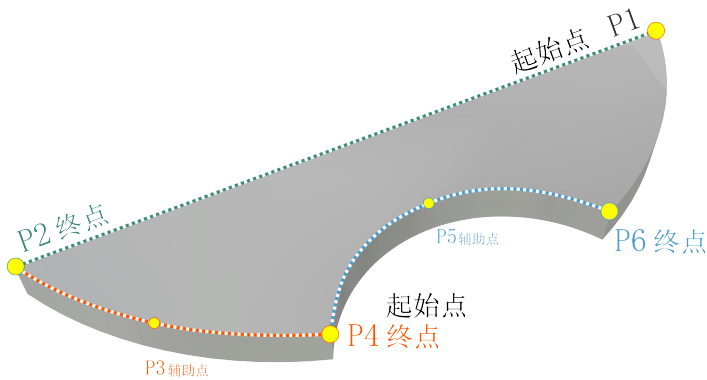


图 7-29：SCIRC 时的两段圆弧

- 1. 将 TCP 移到应被视为 HP 辅助点和 ZP 目标点的位置。

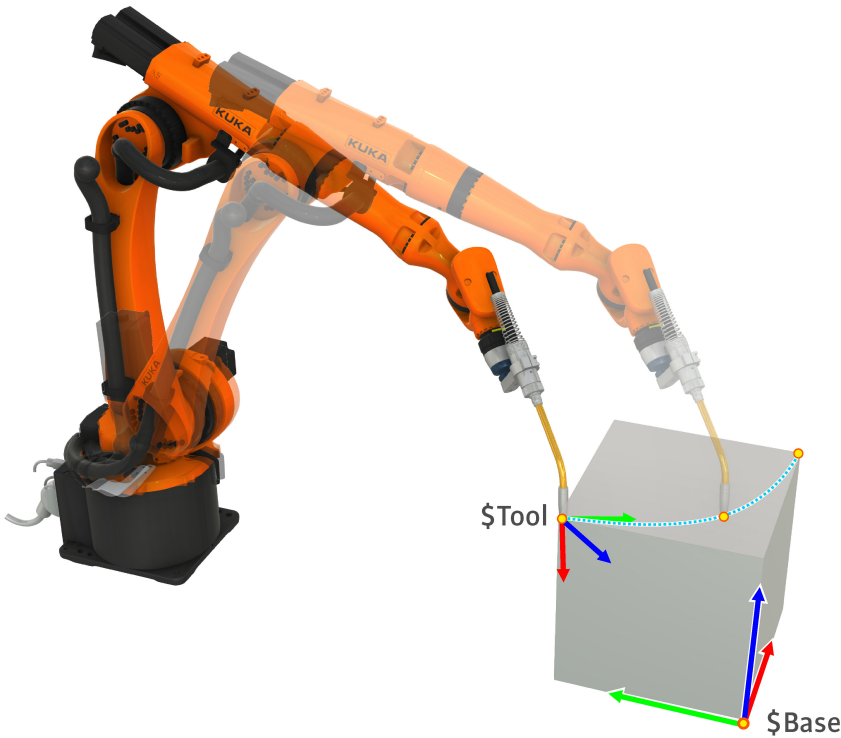


图 7-30：带 SCIRC 的运动指令

- 2. 插入 SCIRC 行指令



图 7-31：插入运动

- 光标选中要在其后面插入运动指令的那一行 (1)。
 - 通过同名按钮插入一个**运动** (2)。
3. 一个预填的行指令将插入在选中的程序行后面。



图 7-32：选择运动方式

- 在行指令中点击运动方式栏 (1)。
 - 一个选择列表打开，在此选择运动方式 SCIRC (2)。
4. 配置各个运动参数。
- SCIRC
(>>> 7.4.3.1 “行指令：SCIRC” 页面 210)
5. 示教辅助点和目标点

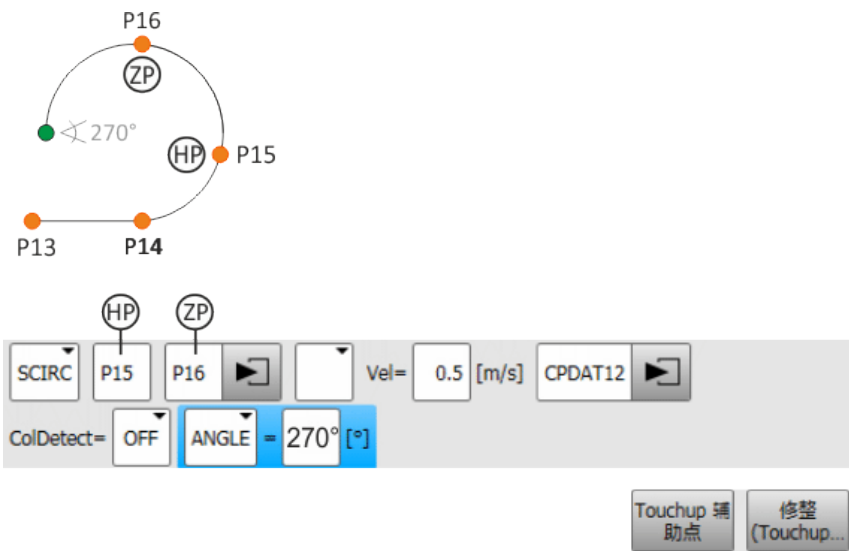


图 7-33：示教 SCIRC 点

- 驶向圆形轨迹上的辅助点（HP）并通过 **Touch-Up 辅助点** 按钮应用该点。
- 驶向圆形轨迹上的目标点（ZP）并通过 **Touch-Up 目标点** 按钮应用该点。



可以选择通过附加参数 **ANGLE**（蓝色栏）将圆形轨迹延伸超出目标点或缩短圆形轨迹。为此需要输入角度。

提示

在更改圆心角时必须检查进一步的轨迹曲线。有碰撞危险

6. 用指令 **OK** 保存指令。

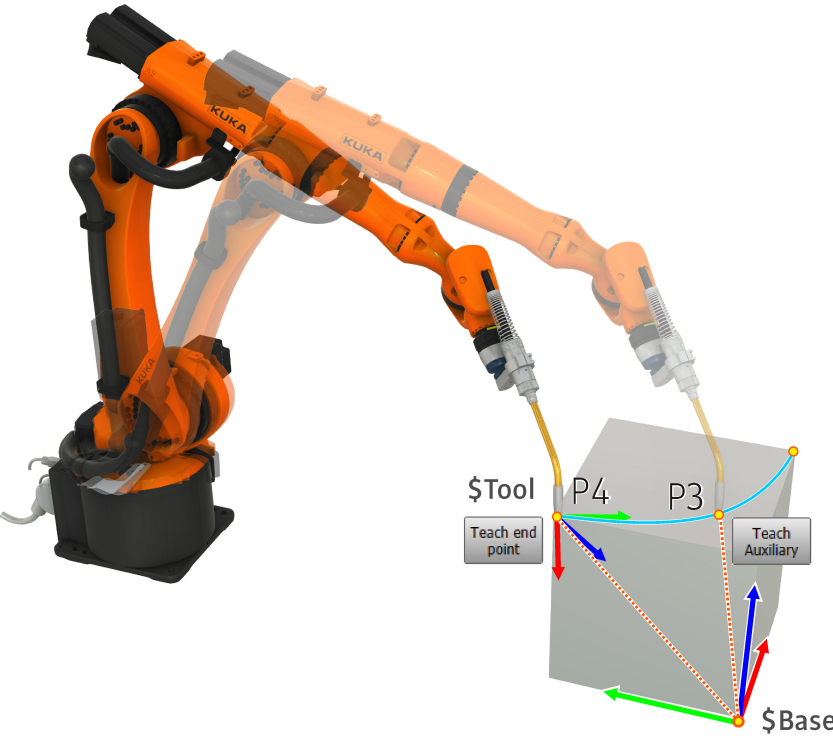


图 7-34：在“Touchup HP”和“Touchup ZP”时保存点坐标

7.4.3.1 行指令：SCIRC

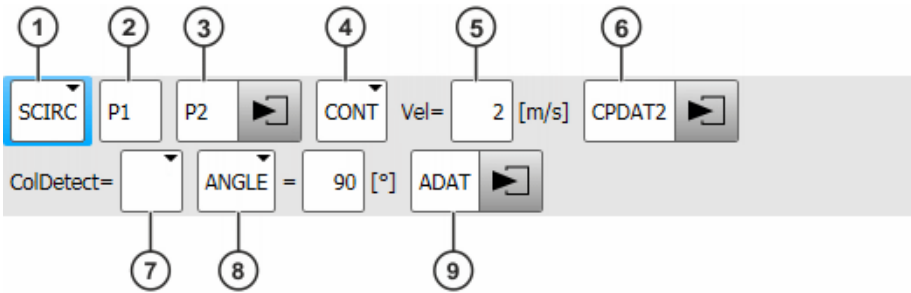


图 7-35：SCIRC 行指令（单个运动）

序号	说明
1	运动方式 SCIRC
2	辅助点名称 系统自动赋予一个名称。名称可以被改写。
3	目标点名称 系统自动赋予一个名称。名称可以被改写。 箭头符号： (>>> “设置工具和基坐标” 页面 196)
4	- CONT：目标点轨迹逼近。 [空]：将精确地移至目标点。
5	速度 0.001 ... 2 m/s
6	运动参数名称。系统自动赋予一个名称。名称可以被改写。 需要编辑点参数时请按箭头。相关选项窗口即自动打开。
7	该运动的碰撞识别 - OFF：碰撞识别已关闭 - CDSets_Set[编号]：碰撞识别已开启。为了识别将使用数据组编号中的数值。
8	圆心角 - 9,999° ... + 9,999° 如果输入的圆心角小于 - 400° 或大于 + 400°，则在保存行指令时会自动询问是否要确认或取消输入。
9	通过切换参数可显示和隐藏该栏目。 含逻辑参数的数据组名称。系统自动赋予一个名称。名称可以被改写。需要编辑参数时请按箭头。相关选项窗口即自动打开。

姿态特性

- 轨迹运动时可以准确定义姿态引导。
- 工具在运动的起点和目标点处的方向可能不同。
 - 工具的姿态特性
(>>> [7.4.3.3 “SCIRC 时的姿态引导” 页面 211](#))
- 工具姿态可能在沿圆形轨迹运行时受其他参数影响：
 - 辅助点上的姿态特性

(>>> 7.4.3.4 “SCIRC: 姿态引导举例: 辅助点” 页面 214)

– 目标点上的姿态特性

(>>> 7.4.3.5 “SCIRC: 姿态特性 - 目标点示例” 页面 216)

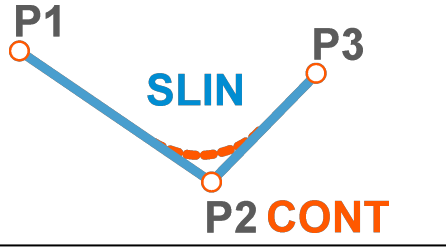
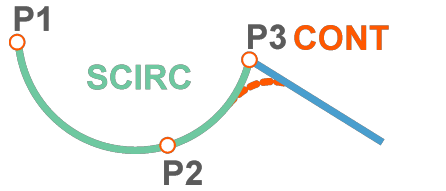
7.4.3.2 轨迹运动时的轨迹逼近

CP 运动的轨迹逼近



轨迹逼近功能不适用于生成圆周运动。它仅用于防止在某点出现精确暂停。

在运行方式 SLIN 和 SCIRC 下进行轨迹逼近

运动方式	特征
	• 轨迹曲线相当于两条抛物线 • 轨迹逼近距离单位为 mm
	• 轨迹曲线相当于两条抛物线 • 目标点轨迹逼近 • 轨迹逼近距离单位为 mm

7.4.3.3 SCIRC 时的姿态引导

姿态引导: 标准或手动 PTP

工具的方向在运动过程中不断变化。在机器人以标准方式到达手轴奇点之前就可以使用手动 PTP，因为是通过手轴角度的线性转换（轴相关运行）进行姿态变化。

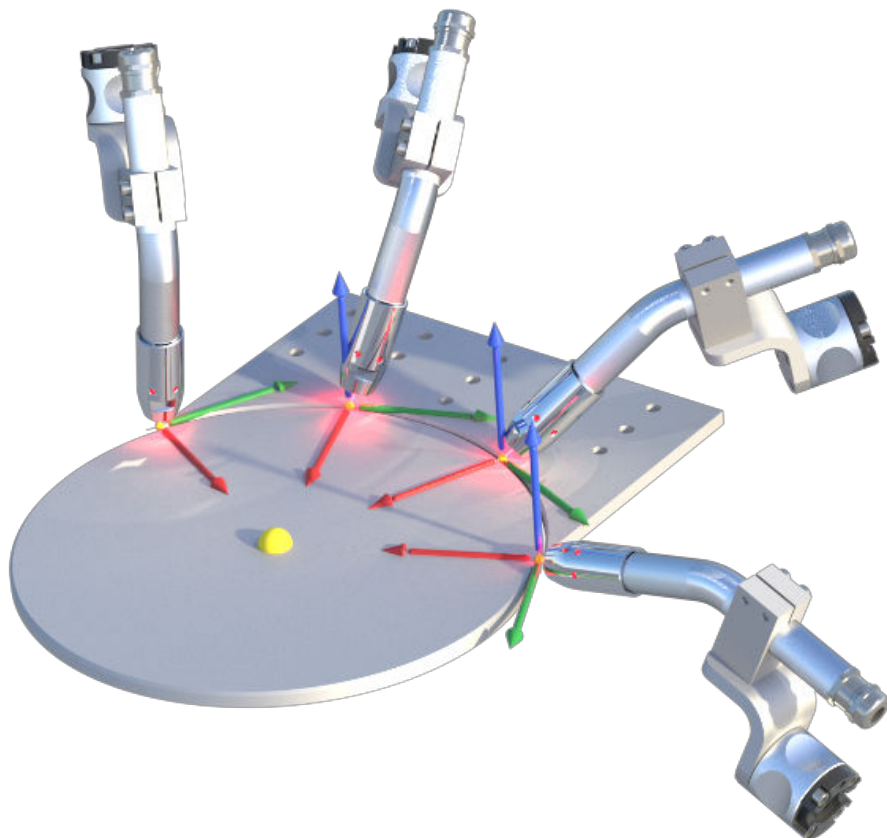


图 7-36: 姿态: 默认 + 基坐标参照系

姿态引导: 恒定不变和基坐标参照系

工具的姿态在运动期间保持不变, 与在起点所示教的一样。在终点示教的姿态被忽略。

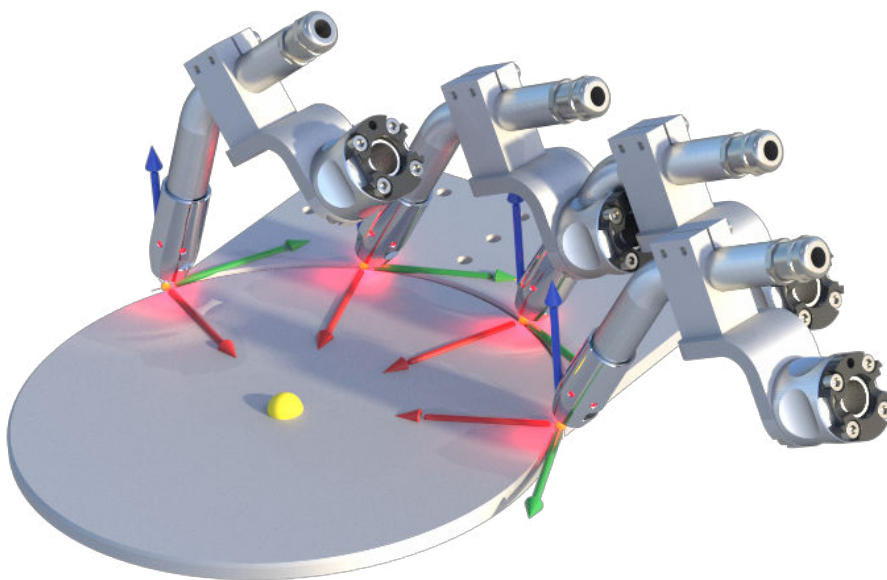


图 7-37: 姿态: 恒定不变 + 基坐标参照系

姿态引导：恒定不变和轨迹参照系

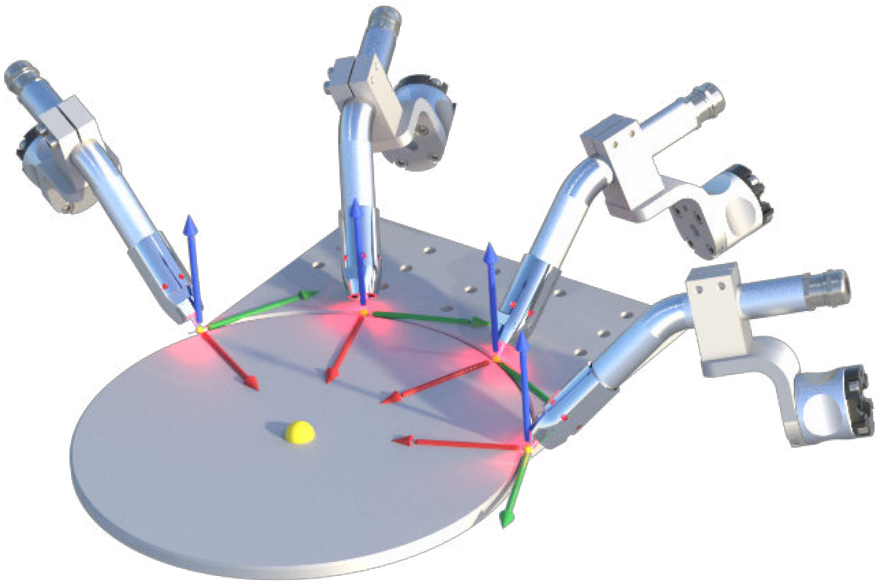


图 7-38：姿态：恒定不变 + 轨迹参照系

操作步骤



图 7-39：运动参数

- 1. 在打开的行指令中点击 CPDATn 栏 (1) 的箭头符号。
- 2. 在显示的弹出窗口中切换到运动参数选项卡 (2)。

参数

3	圆周的姿态引导 - 以基准为参照 - 以轨迹为参照
4	姿态引导 - 标准 - 手动 PTP - 恒定的姿态引导

7.4.3.4 SCIRC: 姿态引导举例：辅助点

初始状态

- 为 TCP 对以下姿态进行了编程：
 - 起始点：0°
 - 辅助点：98°
 - 目标点：197°
- 也就是再定向为 197° 。
如果忽略辅助点，也可通过较短的姿态改变 $360^{\circ} - 197^{\circ} = 163^{\circ}$ 达到目标姿态。
 - 橙色虚线箭头显示已编程的姿态。
 - 如果实际姿态与已编程的姿态有所不同，则灰色箭头以图表形式显示实际姿态如何偏离。

内插 #INTERPOLATE

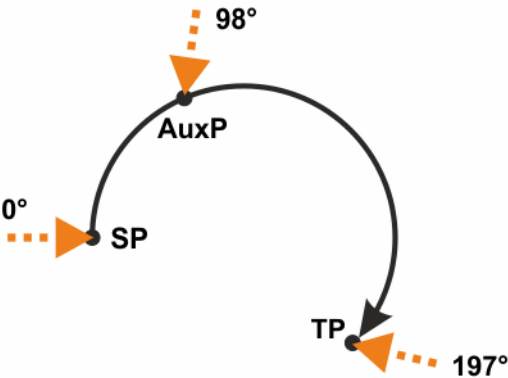


图 7-40: #INTERPOLATE

- SP 起始点
- AuxP 辅助点
- TP 终点

- TCP 在辅助点上接收已编程的姿态 98° 。
- 姿态改变为 197° 。

忽略 *#IGNORE*

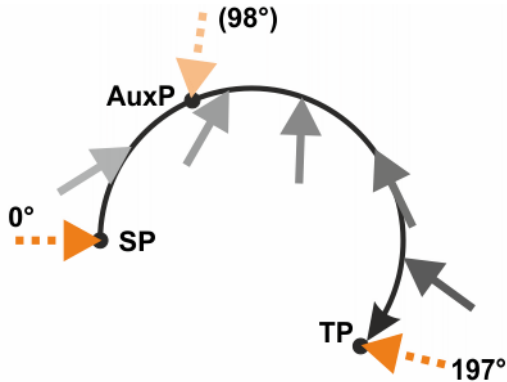


图 7-41: *#IGNORE*

- 以 163° 进行较短的姿态改变。
- 辅助点的编程姿态则被忽略。

考虑 *#CONSIDER*

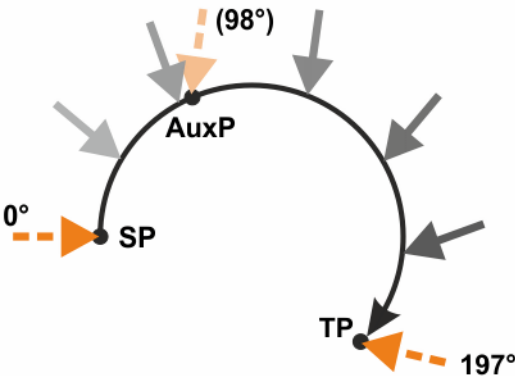


图 7-42: *#CONSIDER*

- 辅助点的编程姿态为 98°，因此姿态位于较长的路程上。
- 所以机器人控制系统为改变姿态选择较长的路程。



用户想要确定如果 TCP 无需到达辅助点的一定姿态，应朝哪个方向改变姿态，那么 *#CONSIDER* 才适用。用户可通过辅助点规定方向。

考虑 *#CONSIDER* 的其他示例

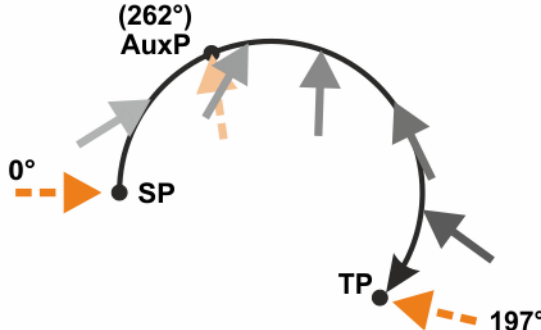


图 7-43: *#CONSIDER*，其他示例

- 如果为辅助点编程 262°，则其位于较短的路程上。
- 所以机器人控制系统为改变姿态选择较短的路程。
- 灰色箭头显示控制系统绝对不会接受辅助点的编程姿态。

操作步骤



图 7-44: SCIRC 姿态特性

1. 在打开的行指令中点击 CPDATn 栏 (1) 的箭头符号。
2. 在显示的弹出窗口中切换到圆周配置选项卡 (2)。
3. 确定辅助点 (3) 和目标点 (4) 的姿态特性。

元素	说明
辅助点上的姿态特性	下拉菜单 • 内插: TCP 在辅助点上应用已示教的姿态。 • 忽略: 机器人控制器忽略辅助点的示教姿态。沿着最短的路径将 TCP 开始姿态转换为目标姿态。 • 考虑 (默认): 机器人控制器选择接近辅助点编程姿态的路径。
目标点上的姿态特性	下拉菜单 • 内插: 在实际的目标点上, 采用编程设定的目标点姿态。 (在没有给出圆心角时, SCIRC 的唯一可能性。如果设定 EXTRAPOLATE, 仍然要执行 INTERPOLATE。) • 外推 (带圆心角数据的 SCIRC 的默认值): 根据圆心角调整姿态: 如果圆心角延长了运动, 则示教目标点上应用已示教的姿态。继续相应调整姿态直至实际目标点。 如果圆心角缩短了运动, 不会达到已应用的姿态。

7. 4. 3. 5 SCIRC: 姿态特性 - 目标点示例

说明

- 橙色虚线箭头显示已编程的姿态。
- 如果实际姿态与已编程的姿态有所不同, 则灰色箭头显示实际姿态。

内插 #INTERPOLATE

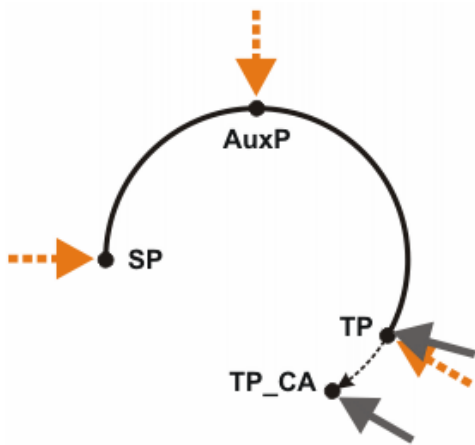


图 7-45: #INTERPOLATE

- SP 起始点
- AuxP 辅助点
- TP 编程的目标点
- TP_CA 实际目标点。通过圆心角得出。

- 在 TP_CA 之前的 TP 中，还没有到达编程设定的姿态。
- 在 TP_CA 上采用编程的姿态。

外插 #EXTRAPOLATE

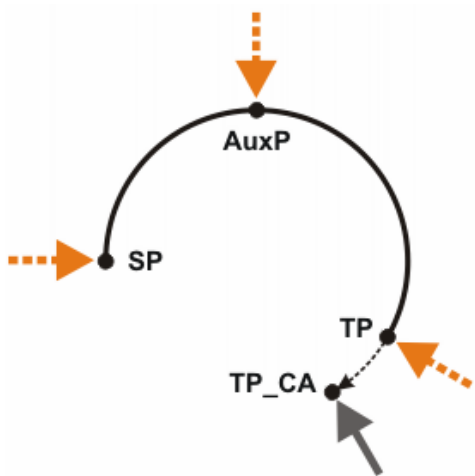


图 7-46: #EXTRAPOLATE

- 在 TP 上采用编程的姿态。
- 根据圆心角为 TP_CA 继续调整姿态。

操作步骤

请参见：
SCIRC: 姿态特性 - 辅助点示例 > 操作步骤
(>>> [图 7-44](#))

7.4.3.6 执行 \$CIRC_MODE 时的限制

限制

系统变量说明的摘要

- 如果 `$ORI_TYPE = #IGNORE` 适用于一个 SCIRC 段，则不分析 `$CIRC_MODE`。
- 如果一个 SCIRC 段在 `$ORI_TYPE = #IGNORE` 适用的 SCIRC 或 SLIN 段之后，则 `#CONSIDER` 不可用于该 SCIRC 段。

用于带圆心角的 SCIRC:

- 不允许为辅助点设定 `#INTERPOLATE`。
- 如果 `$ORI_TYPE = #IGNORE` 适用，不允许为目标点设定 `#EXTRAPOLATE`。
- 如果一个适用于 `$ORI_TYPE = #IGNORE` 的样条段位于前面，不允许为目标点设定 `#EXTRAPOLATE`。

故障信息说明

有些故障信息包含文本“由于违反规则 X 导致故障”

在 `$CIRC_MODE` 编程时必须注意姿态和附加轴:

1. 规则 1: 如果没有忽略起始点和目标点，则正好允许 `#CONSIDER`。
2. 规则 2: Missing inline element 'emphasis'. 如果没有忽略起始点和目标点，则正好允许 `$CIRC_TYPE= #PATH`。
3. 规则 3: 如果已设定 `$ORI_TYPE = #IGNORE` 或 `$EX_AX_IGNORE`，则不再对 `$CIRC_MODE` 进行分析。
4. 规则 4: 如果已为圆心角编程，则禁止辅助点上的插补。
5. 规则 5: 如果已为圆心角编程，没有忽略起始点和目标点，则允许目标点精确通过外推确定。
6. 规则 6: 如果已为圆心角编程，没有忽略目标点，则恰好允许（通过插补）应用目标点。
7. 规则 7: `#CONSIDER` 路径仅在使用无限转动的附加轴时生效。使用不同的轴，始终选择与 `#IGNORE` 相符的较短路径。
8. 规则 8: 如果已为圆心角编程，则仅考虑部件 `TARGET_PT`。
9. 规则 9: 所有位置均不允许读入 `$CIRC_MODE`，仅可在 SCIRC 的 WITH 令牌中写入。

7.4.4 沿轨迹运行和轨迹逼近

练习内容

打开课程练习本并完成以下练习：

- 沿轨迹运行和轨迹逼近

7.5 用全局点编程

说明

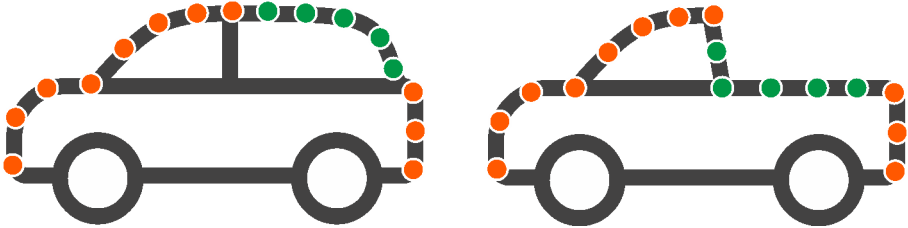


图 7-47：全局和局部点

- 在上例中，必须将点移向两个类似的车辆。
 - 橙色点在两个车身上相同。
 - 绿色点在两个车身上不同。
- 为两辆车预定了不同的程序。
- 为了最小化编程和维护耗时，
 - 将橙色点示教为全局点。
 - 在程序数据列表中将绿色点示教为局部点。
- 在控制系统的单独全局点数据列表中保存全局点。



示教全局点功能必须已由管理员激活。

权限管理



为了较低级别的用户组也可以使用示教全局点功能，管理员必须在权限管理中解锁该功能。



图 7-48：权限管理

限制

全局点可用于以下运动

- SLIN （既可以在样条组中也可以在单个语句中）
- SPTP （既可以在样条组中也可以在单个语句中）
- LIN
- PTP

全局点无法用于

- 作为整体的样条组

- SCIRC
- CIRC



起始位置 (HOME) 已经是全局点！

操作步骤



图 7-49：激活全局点

1. 在行指令中打开点参数 (1)。
2. 切换到参数选项卡 (2) 中。
3. 通过同名按钮激活全局点 (3)。
4. 在打开的行指令中通过附加 GLOBAL 标记一个全局点

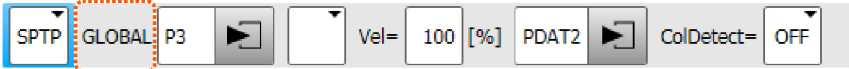


图 7-50：行指令中的全局点

5. 用按钮 OK 应用点。
6. 在编辑器中，全局点的名称始终放在 <> 尖括号中。

• 局部保存的点

1 SPTP P3 Vel = 100%

• 全局保存的点

1 SPTP <P3> Vel = 100%



可以为每个点单独决定要将其保存为局部点还是全局点。

提示

如果要通过 Touch-Up 重新更改一个全局点的位置，则系统指出可能重复使用/碰撞危险。

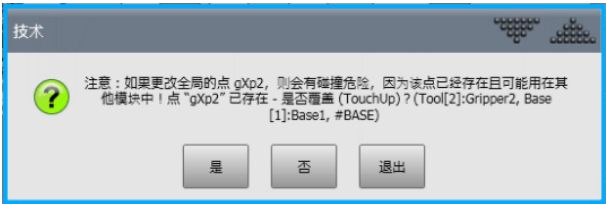


图 7-51：提示信息

Global_Points.dat

- 在 Global_Points.dat 中保存在行指令中全局激活的点。
- 菜单路径: KUKA_DISK C: > KRC > Roboter (机器人) > KRC > R1 > System (系统) > Global_Points.dat

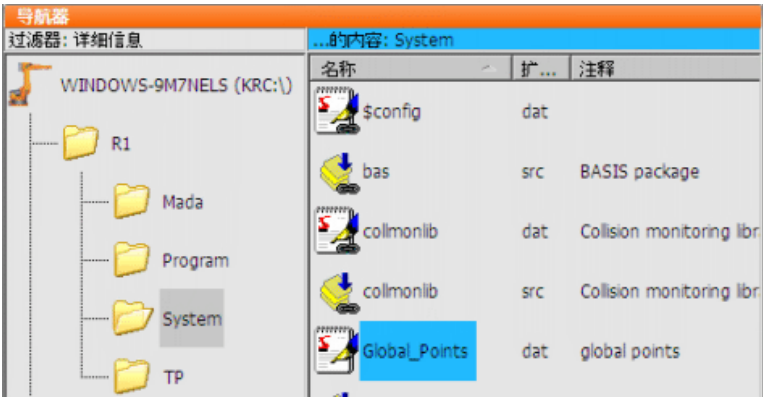


图 7-52: Global_Points.dat

- 示例: 在 Global_Points.dat 中保存的点

```
DEFDAT GLOBAL_POINTS PUBLIC

DECL GLOBAL E6POS gXP1={X 1765.00,Y -2.81284883E-07,Z 1784.00,A 0.0,B 90.0000,C 9.13
DECL GLOBAL FDAT gFP1={TOOL_NO 1,BASE_NO 0,IPO_FRAME #BASE,POINT2[] " "}
DECL GLOBAL PDAT gPPDAT1={VEL 100.000,ACC 100.000,APO_DIST 100.000,APO_MODE #CPTP,GEA

ENDDAT
```

- 句法:
 - 局部数据列表中通过行指令示教的点。
 - Global_Points.dat 中通过行指令示教的点。

```
xP1 = {X 1700, Y 23, Z 1200, A 0, B 90, C 45}
```

```
gxP1 = {X 1700, Y 23, Z 1200, A 0, B 90, C 45}
```

i 由于前缀 xPn 和 gxPn 不同，控制系统上的每个点都是唯一的。

7.5.1 在概览中显示和更改全局点

说明

可在任何程序模块中生成全局点。可通过 smartPAD 调用“全局点”概览，其中列出了所有存在的控制系统全局点。在概览中通常可编辑 Global_Points.dat 中包含的点。

操作步骤

- 在主菜单中选择配置 > 全局点。

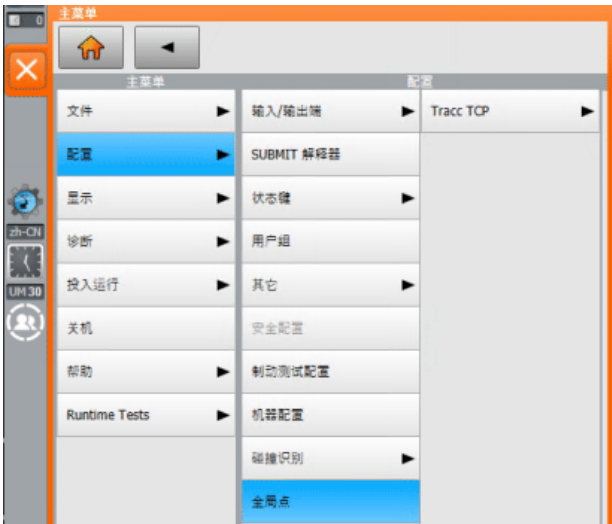


图 7-53：菜单路径，全局点

2. 全局点窗口打开。



图 7-54：点概览，搜索

3. 搜索全局点，必要时更改。
4. 点击**保存**。即显示一个安全询问。
5. 若用**是**回答此安全询问，将保存更改。

全局点窗口



图 7-55: 全局点窗口

序号	说明
1	显示其全局点的文件 默认情况下此处仅有 Global_Points.dat 可用。可添加其他文件。
2	输入要用来筛选显示的名称或名称片段
3	将所有全局点存档在 C:\KRC\Roboter\LOG\GlobalPointsLogFile.txt 中。（无论序号 1 下可能选择了哪个文件。）
4	点列表，并给出变量名称和数据类型 可通过箭头符号展开列表条目。
5	展开的条目，显示点数据
6	在编辑模式下打开列表条目。
7	显示使用相应全局点的所有文件的列表。

示例：用途概览

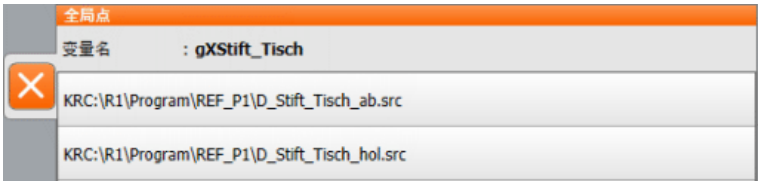


图 7-56: 全局点，使用

7.6 更改运动命令

更改运动指令

更改现有运动指令的原因有多种：

典型原因	待执行的更改
- 待抓取工件的位置发生变化。 - 加工时五个孔中的一个孔位置发生变化。 - 焊缝必须截短。	位置数据的更改
货盘位置发生变化。	更改坐标系数据：Base
意外使用了错误的基坐标系或错误的工具坐标系对某个位置进行了示教。	更改坐标系数据：基坐标系和/或工具坐标系并更新位置
加工过缓慢：必须改进节拍时间。	- 更改运动数据：速度、加速度 - 更改运动方式

更改位置数据的作用

只更改点的数据组：用“Touch-Up”更新数值，因此点获得新的坐标。

提示
旧的坐标会被覆盖并且不可恢复。

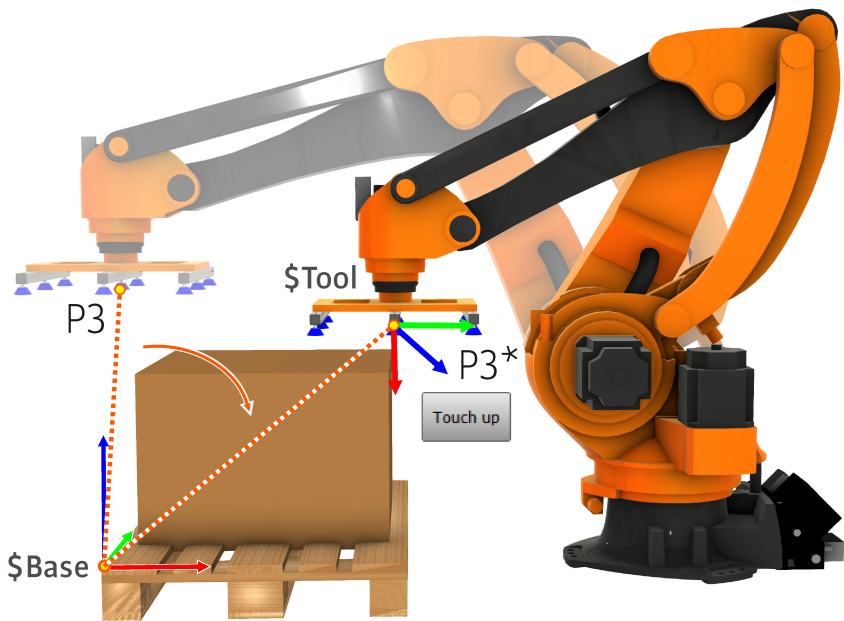


图 7-57：用“Touchup”更改机器人位置

操作步骤

1. 设置运行方式 T1，将光标放在要改变的指令行里。
2. 将机器人移到新位置。

3. 点击**更改**。



图 7-58: 行指令, 按键

指令相关的行指令将打开。

4. 对于 SPTP 和 SLIN 运动:

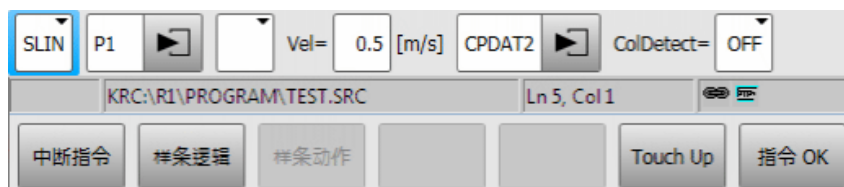


图 7-59: 编辑行指令

- 按下 **Touchup**，以将 TCP 的当前位置作为新的点。

对于 SCIRC 运动:



图 7-60: SCIRC, 按键

- 按 **Touchup HP**（辅助点坐标），以便确认 TCP 的当前位置为新的辅助点。
- 或者按 **Touchup ZP**（目标点坐标），以便确认 TCP 的当前位置为新的目标点。

5. 对话框: “确实要记录“XPnn”点吗?”



图 7-61: 信息

点击**是**确认。

6. 用命令 **OK** 保存更改。
7. 在程序中测试点是否会发生碰撞。

更改坐标系数据的作用

更改坐标系数据（例如工具、基坐标）时，会导致位置发生位移（比较“矢量位移”）。机器人位置会发生变化！（与机器人的轴角度位置相符）。旧的点坐标依然会被保存并有效。发生变化的仅是参照系（例如基坐标）。可能会出现超出工作区的情况！因此机器人不能到达某些位置。如果机器人位置保持不变，但坐标系参数改变，则必须在更改参数（例如基坐标）后在所需位置上用“Touch-Up”更新坐标！

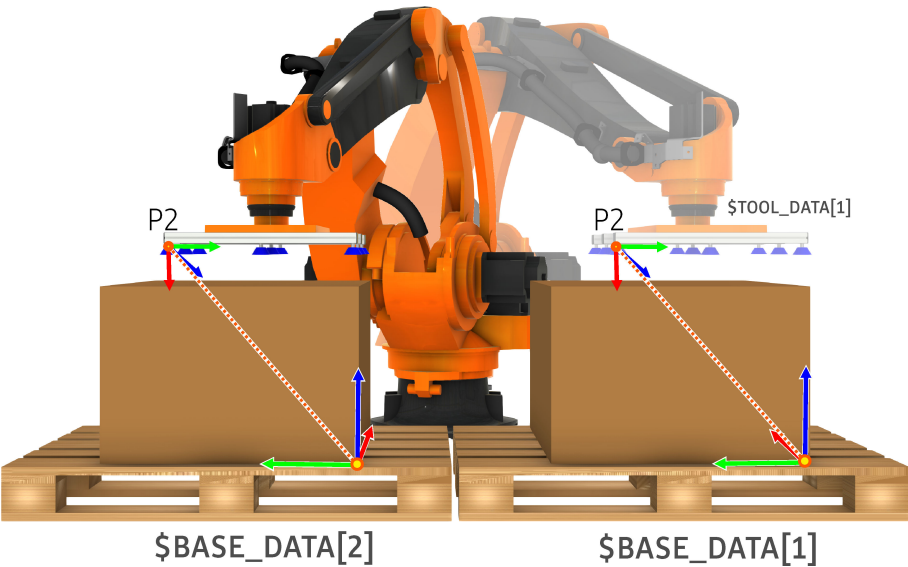


图 7-62：更改 FRAME 参数（以基坐标为例）



此外，用户对话框会发出警告：“注意：更改以点为参照的坐标系参数时会有碰撞危险！”

操作步骤

- 1. 将光标放到要更改的指令行中。
- 2. 点击**更改**。



图 7-63：行指令，按键

指令相关的行指令将打开。

- 3. 打开选项窗口“坐标系”。
- 4. 设置新工具坐标系或者基坐标系或者外部 TCP。

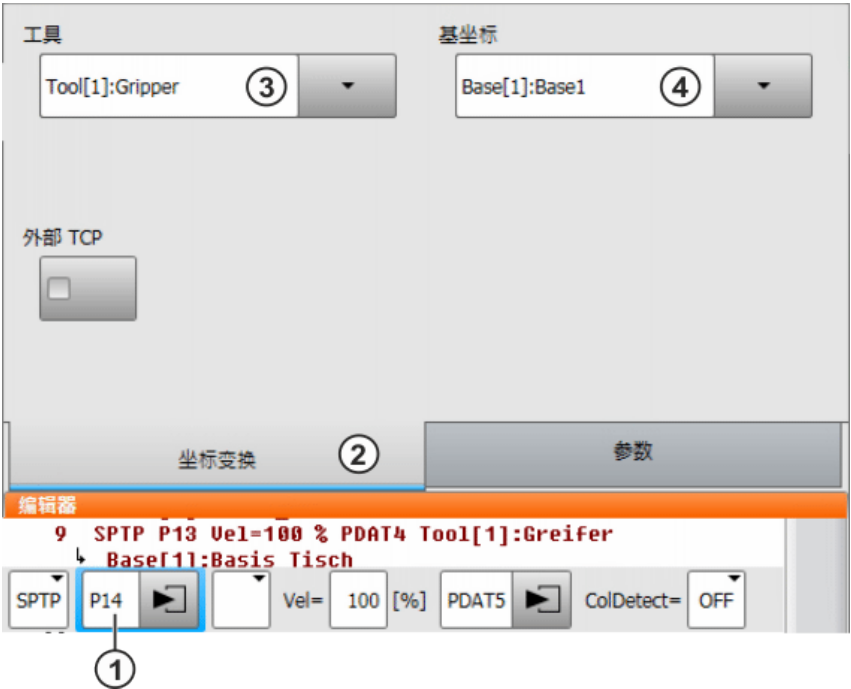


图 7-64：设置基坐标和工具

5. 用 **OK** 确认用户对话框“注意！改变以点为基准的坐标系参数时会有碰撞危险！”。



图 7-65：信息，碰撞危险

6. 如要在工具和/或基坐标设置已更改的情况下**保留当前机器人位置**，则必须按下 **Touch Up**，以便重新计算和保存当前位置。必须事先已移至要保持的位置。
7. 用**指令 OK** 保存更改。



如果坐标系参数发生变化，则必须重新测试程序是否会发生碰撞。

更改运动数据的作用

更改速度或者加速度时会改变移动属性。这可能会影响加工工艺，特别是使用轨迹应用程序时：

- 胶条厚度
- 焊缝质量

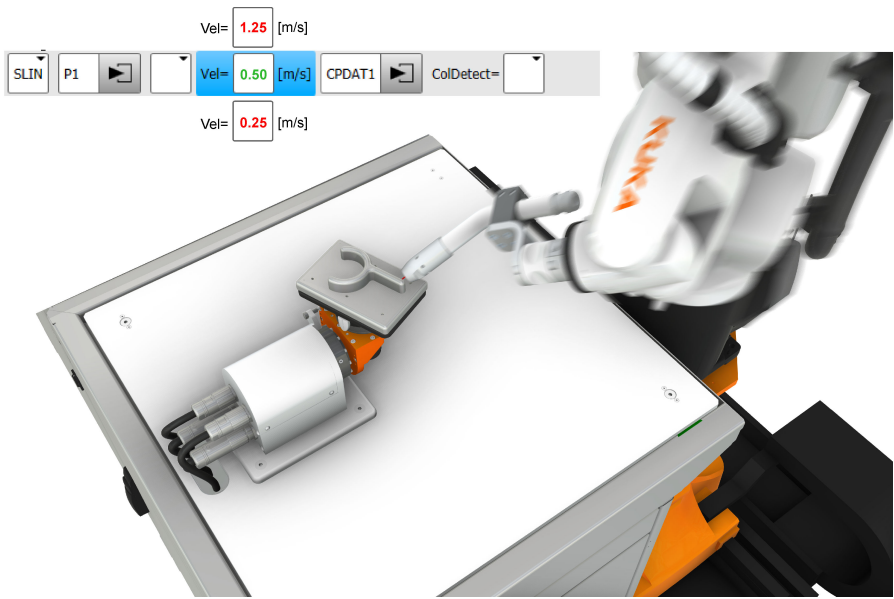


图 7-66：更改运动数据（以速度为例）

更改运动方式的作用

更改运动类型时总是会导致轨迹规划更改！这在不利情况下可能会导致发生碰撞，因为轨迹可能会发生意外变化。

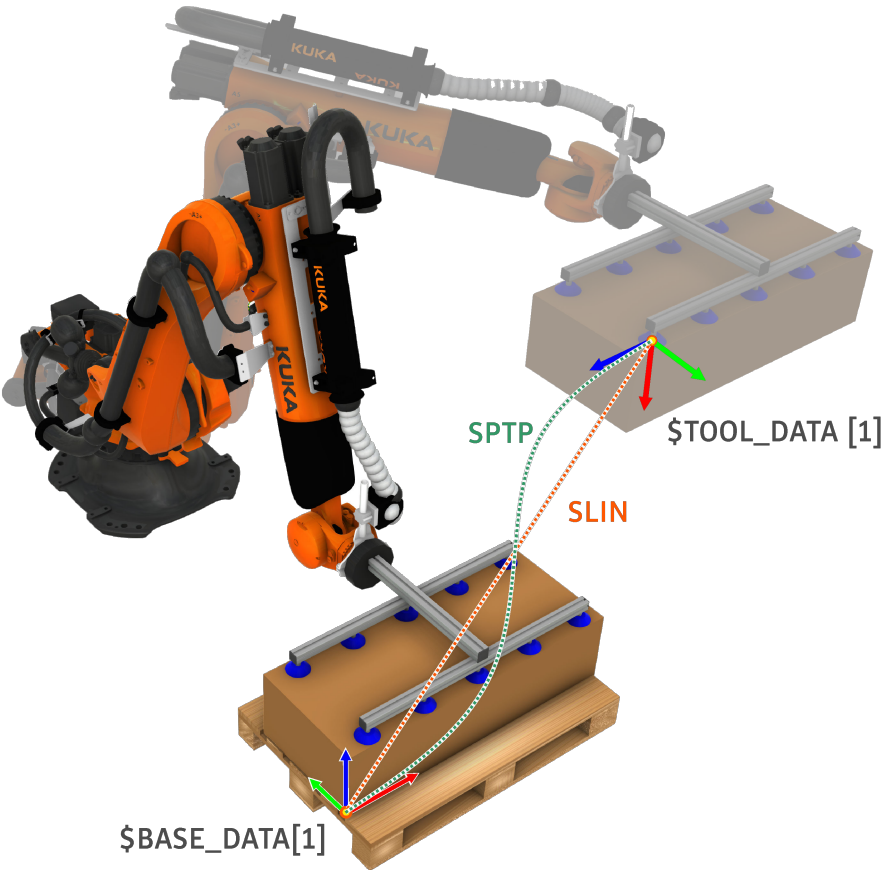


图 7-67：更改运动方式

操作步骤

1. 将光标放到要更改的指令行中。
2. 点击更改。



图 7-68：行指令，按键

指令相关的行指令将打开。

3. 更改参数。
4. 用**指令 OK** 保存更改。



更改运动参数后必须重新检查程序是否会引发碰撞及过程是否安全。

这种方法可用于以下更改：

- 运动方式
- 速度
- 加速度
- 轨迹逼近
- 轨迹逼近距离

关于更改运动指令的安全提示

提示

- 每次更改完运动指令后都必须在低速（运行方式 T1）下测试机器人程序。
- 立即以高速启动机器人程序可能会导致机器人系统和整套设备损坏，因为可能会出现不可预料运动。
- 如果有人位于危险区域，则可能会造成重伤。

点的重命名

1. 用光标选定待重命名的点。
2. 用按键“更改”打开行指令。
3. 在行指令中修改点名称（点名称栏）。
4. 用按键“指令 OK”应用更改。
5. 出现以下对话框：为“点名称”保留之前的点坐标？(TOOL_DATA[n], BASE_DATA[n], #BASE)
6. 通过软键“是”应用原点的先前坐标。

点的多重使用

1. 在所需的程序位置插入新的行指令。
2. 在行指令中输入可反复使用的点名称（点名称栏）。
这里请务必注意原来的点名称的具体书写形式。
3. 进行其他设置并且用按键“指令 OK”应用所作的设置
4. 出现以下对话框：点“点名称”已存在 - 是否覆盖？(TOOL_DATA[n], BASE_DATA[n], #BASE)
5. 使用软键“否”不覆盖原点的现有点参数。

插入程序行

1. 用光标选定现有的程序行。
2. 用按键“移动”打开新的行指令。



总是在选定的程序行后插入新的行指令。后续程序行向后移动一行。

删除程序行

- 1. 用光标选定待删除的程序行
- 2. **菜单路径:** 编辑 > 删除



点的值在相应的 .dat 文件中保持不变。

8 样条运动编程

8.1 学习单元：样条运动编程

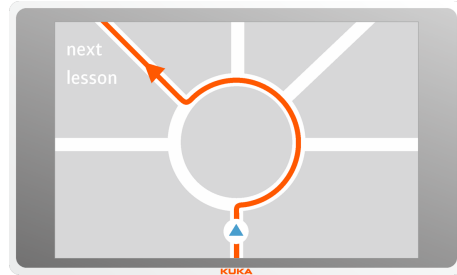


图 8-1：导航器

本学习单元讲授以下内容：

- 什么是样条？
- 样条组的说明
- 编程带运动的样条组
- 编程提示
- 样条曲线的轨迹逼近

8.2 什么是样条？

说明

借助单个运动编程

- 机器人程序首先由单独的示教点或计算出的点组成。

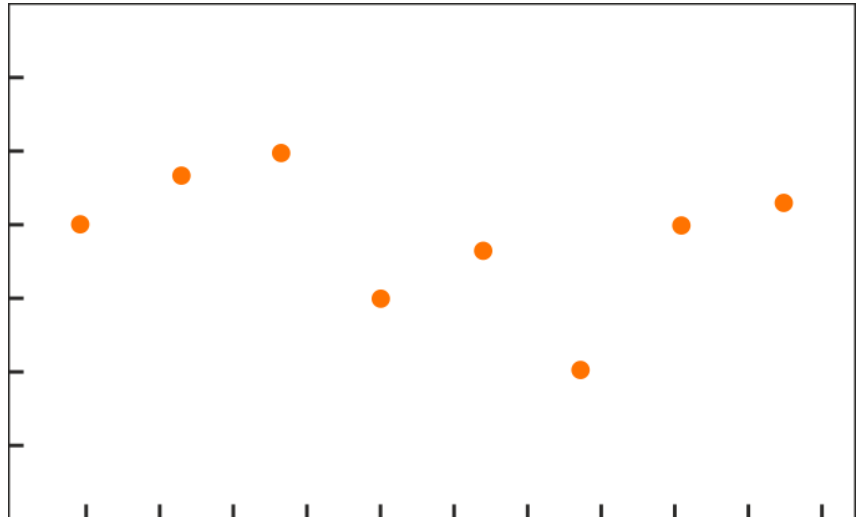


图 8-2：初始情况位置

- 另外还通过行指令或 KRL 编程告诉机器人这些点应该如何相互连接。
- 示例：这里可能的运动语句是 (S)LIN。» 点将通过线段相互连接。

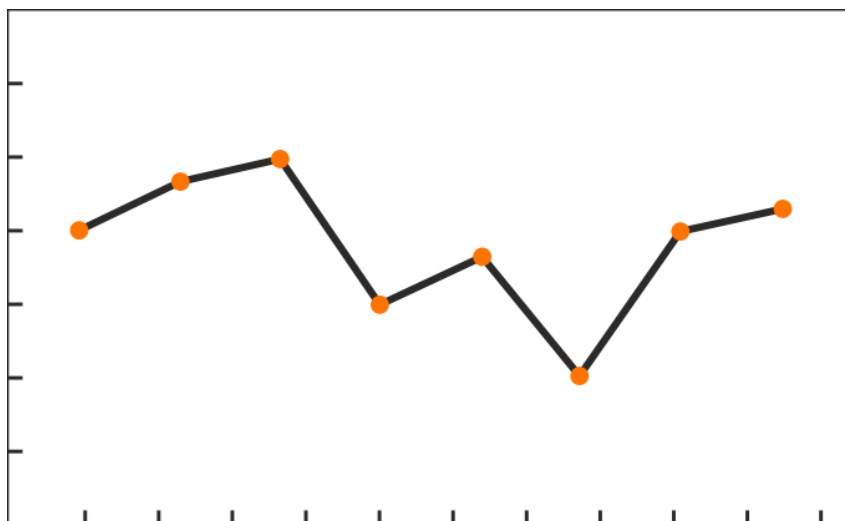


图 8-3：线性插补



缺点：机器人在每个点上都会停止，然后重新加速。

- 为了避免精确停止，可以使用 CONT 参数轨迹逼近这些点。

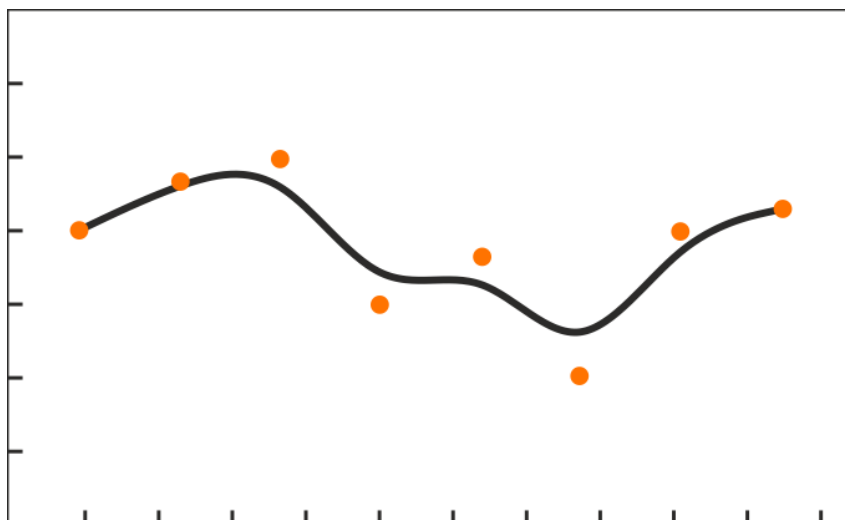


图 8-4：B-SPLINE 插补



缺点：机器人轨迹逼近示教点/计算出的点。

轨迹上只有起点和终点。

优势：机器人以流畅的运动驶过整个轨迹。



确切地说，这是样条的一个特殊形式。

什么是样条？

- **拼板板条**
 - 术语“样条”（SPLINE）可以追溯到绷紧“拼板”的木板，以前在造船和飞机制造中借此确定平滑的肋骨轮廓。

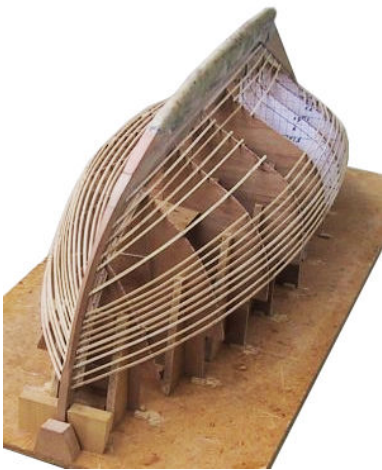


图 8-5：船体，木板结构

- 在这种方法中，通过放置在插补点（图中为钉/销）的拼板重块迫使拼板板条（弹性尺，英文为 spline）满足插补条件（轮廓）。

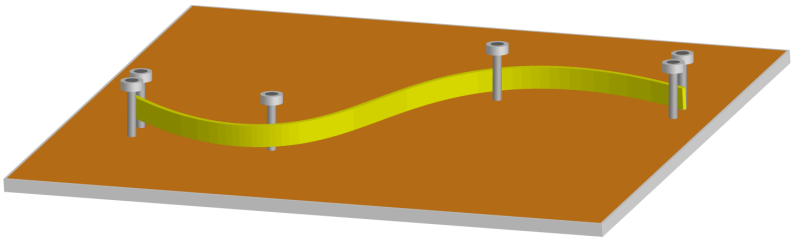


图 8-6：拼板板条

- 然后由于材料的弹性而产生总曲率最小的弯曲线。
- 节点被移动

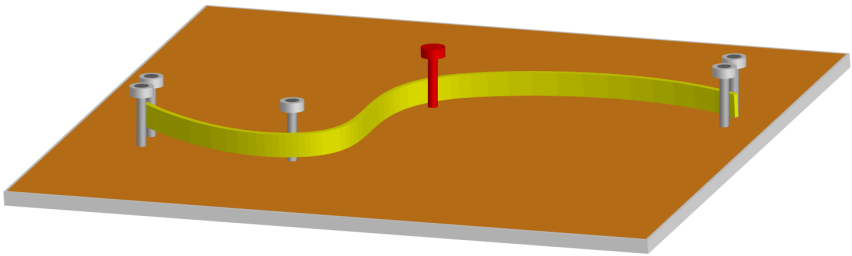


图 8-7：移动节点

- 移动节点（红色钉/销）会影响拼板板条的总轨迹。
- 通过多个节点可以精确指定轨迹。

数学背景

- 多项式插值

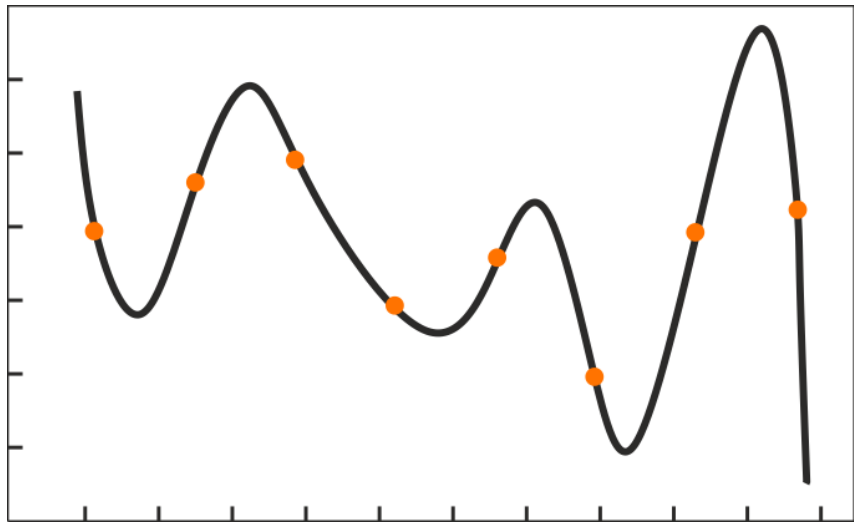


图 8-8：多项式插值

- 多项式插值，通过点的 n 次函数
- 例如如果使用更高次的多项式函数作为插值函数（见图），则可能导致曲线**极端振荡**。
- 从 8 个给定的位置创建一个 7 次多项式 ($ax^7+bx^6+\dots$)



然而在诸如造船或飞机工业的设计应用中，非常不希望出现这种剧烈振荡，这里只需要各个节点之间略微弯曲的平滑曲线（想象一下锯齿形船体…）。

• 样条插补

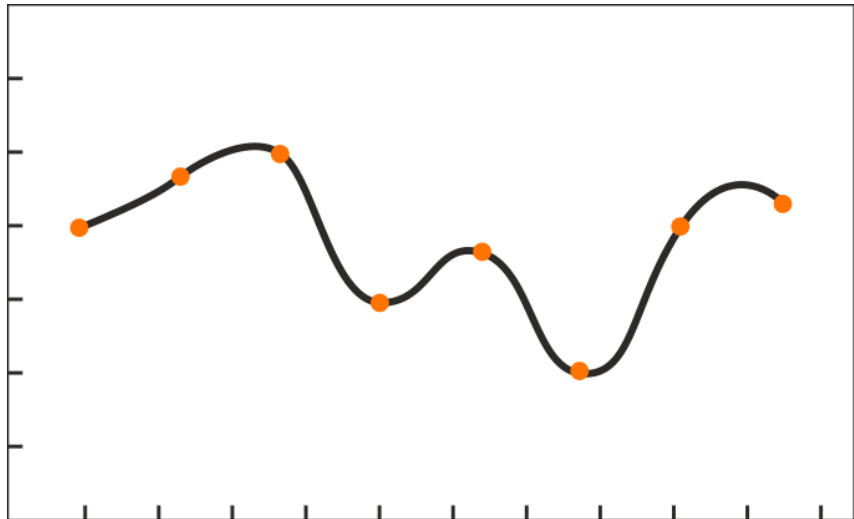


图 8-9：样条插补

- 样条不会尝试通过所有节点插入函数，而是仅在本段通过两个节点进行插值。
- 这些样条具有平滑效果，并且只有非常小的波度。因此它们可以更逼真地表示轮廓。
- 此时最常使用通过线性函数或者 3 次或 4 次多项式 (**KUKA**) 进行插值的线性或三次样条。



另见 (>>> [图 8-4](#))

8.3 样条组的说明

说明

除了带 SPTP、SLIN、SCIRC 的单个语句之外，还有一个“样条组”可供使用。样条组是一种尤其适用于复杂曲线轨迹的运动方式。样条组被看作是一个复杂的轨迹运动进行轨迹规划。



图 8-10

样条组分为两种：

- **样条组**
CP 运动样条组（SPL、SLIN、SCIRC）
- **PTP 样条组**
轴运动样条（只针对 SPTP）

轨迹根据所有的点进行规划，并且经过所有的点。完全预先计算样条组的轨迹。

由于机器人轴限制的原因，在样条轨迹轮廓非常紧凑的情况下始终会引起速度下降。

样条轨迹

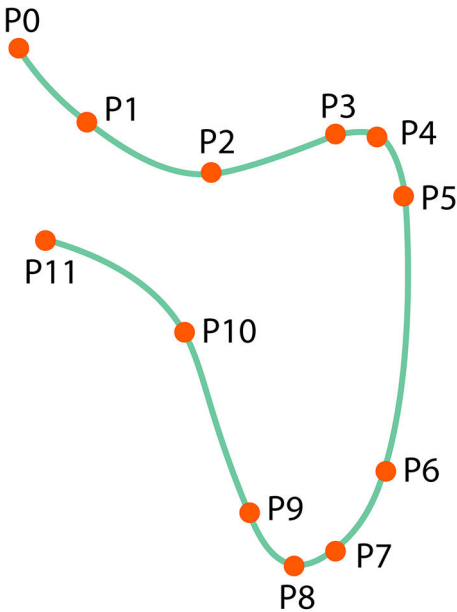


图 8-11：曲线轨迹，样条组

轨迹通过轨迹经过的节点定义。机器人根据节点计算轨迹在样条组中，此外还可以定义专门的匀速运动区域。轨迹曲线始终保持不变，不受倍率、速度或加速度的影响。可以精确地沿圆周和小圆弧运行。

样条组的结构

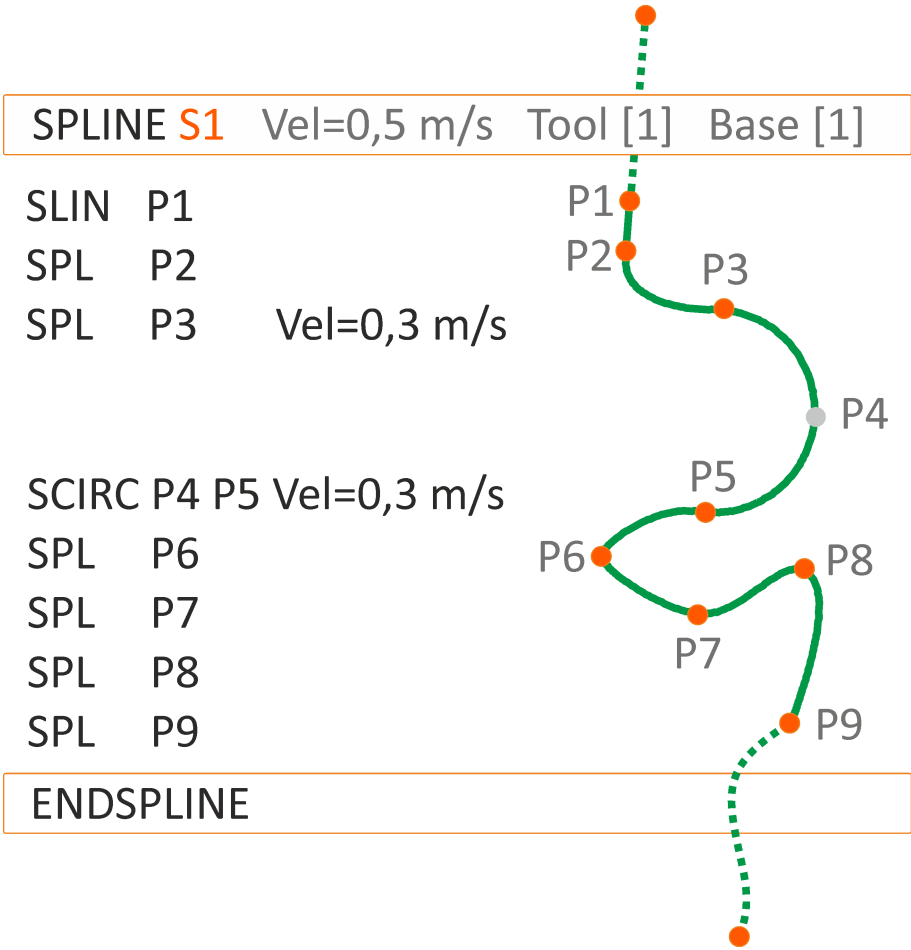


图 8-12：样条组

- 一个样条组是一个由控制系统作为运动插入的折叠。
- 在该折叠中将示教作为样条运动插入的节点。
- 另外，可以在各个样条段中强制机器人进入直线或圆形轨迹。这通过在段中选择朝向节点的运动方式来实现。
- 由此产生用于各个样条段的下列运动方式：
 - SLIN
 - SCIRC
 - SPL
- 对于整个样条组，在“标题行”中确定运动名称和全局参数。
 - 工具、基坐标
 - 速度
 - 轨迹逼近
 - 运动参数
 - 碰撞识别打开/关闭
- 在每个单独示教的运动中可能有所偏差。



例外：通过“标题行”确定工具和基坐标。

- 除了运动指令以外，一个样条组还允许包含以下内容：
 - 提供支持样条组的软件包行指令。
 - 注释和空行
- 可在行指令内集成样条逻辑（触发器、条件停止、匀速）。
样条组不允许含有其他指令，例如变量赋值或逻辑指令。



样条组之前和之后的点确定进入和离开样条组时的轮廓。



样条组的起始点是该样条组前的最后一个点。
样条组的目标点是该样条组中的最后一个点。
一个样条组不会触发预运行停止。

8.4 借助运动对样条组编程

1. 插入样条组



图 8-13：插入样条组

- 将光标放到其后应插入样条组的程序行（1）上。
 - 点击指令按钮（2）。
 - 打开运动条目（3）。
 - 在弹出菜单中选择样条组（4）。
2. 样条组即被插入。

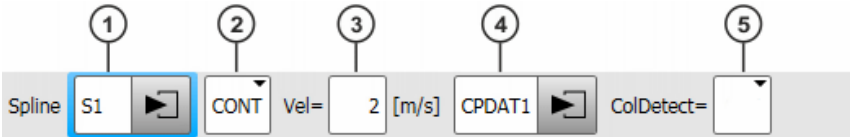


图 8-14：样条行指令

- 确定样条组的名称（1）。
- 所有其他参数都对应于单句编程中的参数，适用于整个样条组。
（工具、基坐标、轨迹逼近、TCP 中的速度、姿态引导、碰撞识别、轴速度和加速度）



在每个单独示教的点上可能与这些参数有偏差。

3. 样条组从编程技术来说是一个折叠。在该折叠中示教要驶过的点。



图 8-15: 打开折叠

- 如果新插入一个样条组，折叠就已经打开。
- 在长时间无动作后折叠会自动关闭，必须重新打开。
- 为此选中样条组（1），然后用同名按钮打开/关闭折叠（2）。

4. 将机器人运行到要新接受的点。

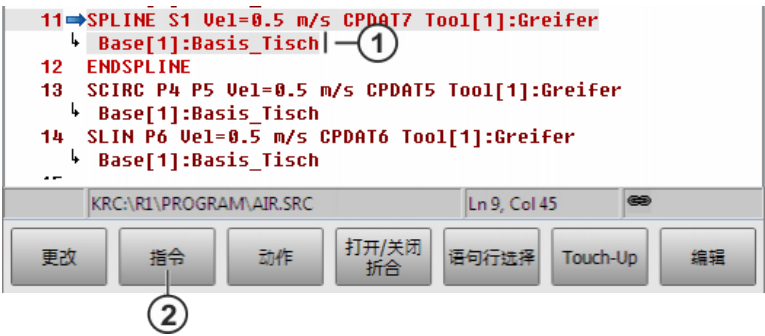


图 8-16: 接受点

- 将焦点置于样条组（1）中。
对于已编程的点，是指要插入新点的位置。
- 使用运动按钮（2）在样条组中插入新点。

5. 插入运动



图 8-17：运动

- 菜单路径：指令 > 运动 > SLIN、SCIRC 或 SPL 可选：

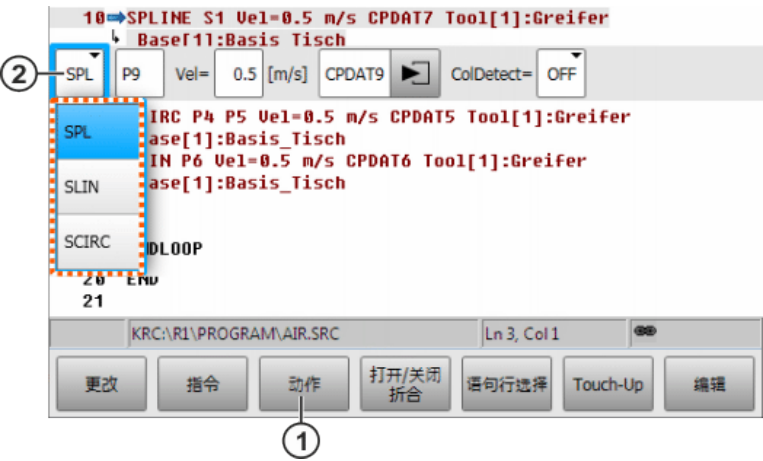


图 8-18：插入运动

- 在打开的折叠中通过同名按钮插入**运动**（1）。
- 通过下拉窗口（2）更改所需的运动。

8.4.1 样条组中的 SLIN

说明

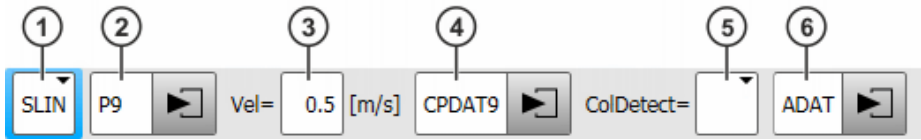


图 8-19: SLIN 段行指令

序号	说明
1	运动方式 SPL、SLIN 或 SCIRC
2	运动点 - 系统自动分配的名称可以更改。 - 示教的点坐标将保存在这个点上。
3	笛卡尔速度 - 默认情况下，对样条组有效的全局值适用于该运动。 - 输入的值适用于该运动 0.001 … 2 m/s
4	运动参数 - 系统自动分配的名称可以更改。 - 默认情况下，对样条组有效的全局值适用于该运动。 - 输入的值适用于该运动 - 需要编辑参数时请按箭头。 运动参数窗口随即打开：（>>> 8.4.4 “运动参数” 页面 246）
5	该运动的碰撞识别 - OFF：碰撞识别已关闭 - CDS_{Set_Set}[编号]：碰撞识别已开启。为了识别将使用数据组编号中的数值。
6	逻辑参数 - 系统自动分配的名称可以更改。 - 输入的值适用于该运动 - 需要编辑参数时请按箭头。 - 相关选项窗口即自动打开。 逻辑参数 — 触发器 — 条件停止 — 恒速

切换参数



图 8-20：切换参数

- 默认情况下不会显示行指令的所有栏。
- 通过切换参数按钮可以在行指令打开时显示和隐藏附加参数。

8.4.2 样条组中的 SCIRC

说明

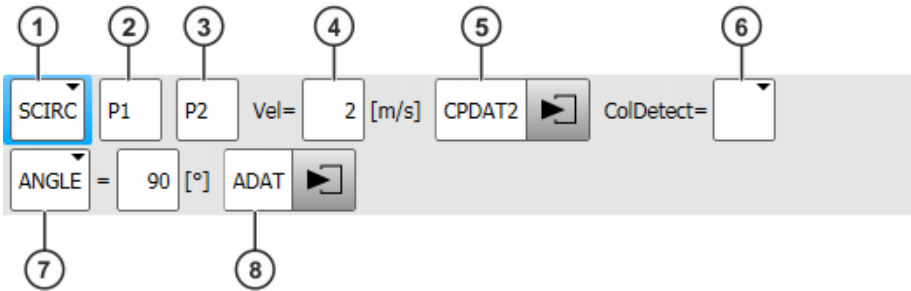


图 8-21：SCIRC 样条段行指令

序号	说明
1	运动方式 • SPL、SLIN 或 SCIRC
2	辅助点 • 系统自动分配的名称可以更改。 • 示教的点坐标将保存在这个点上。
3	目标点 • 系统自动分配的名称可以更改。 • 示教的点坐标将保存在这个点上。
4	笛卡尔速度 • 默认情况下，对样条组有效的全局值适用于该运动。 • 输入的值适用于该运动 • 0.001 … 2 m/s

序号	说明
5	运动参数 - 系统自动分配的名称可以更改。 - 默认情况下，对样条组有效的全局值适用于该运动。 - 输入的值适用于该运动 - 需要编辑数据时请触摸箭头。 运动参数窗口随即打开： (>>> 8.4.4 “运动参数” 页面 246)
6	该运动的碰撞识别 - OFF：碰撞识别已关闭 - CDS_{Set_Set}[编号]：碰撞识别已开启。为了识别将使用数据组编号中的数值。
7	圆心角 - 可选择激活，以便相对于目标点行驶较大或较小的圆形轨迹。 - 9 999° ... + 9 999° - 如果输入的数值小于 - 400° 或大于 + 400° ，则在保存行指令时会自动查询是否要确认或取消输入。
8	逻辑参数 - 系统自动分配的名称可以更改。 - 输入的值适用于该运动 - 需要编辑数据时请触摸箭头。 - 相关选项窗口即自动打开。 逻辑参数 — 触发器 — 条件停止 — 恒速

8.4.3 样条组中的 SPL

说明

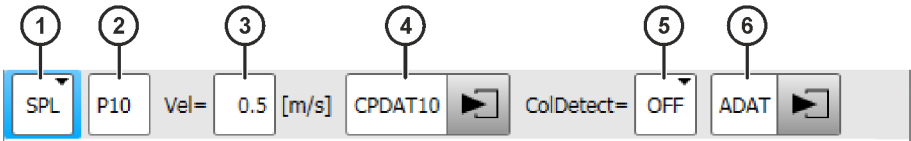


图 8-22：SPL 段行指令

请参见：样条组中的 SLIN
(>>> 8.4.1 “样条组中的 SLIN” 页面 243)

8.4.4 运动参数

运动参数



图 8-23： 运动参数

1	轨迹逼近距离 只有在行指令中选择了 CONT 之后，此栏才显示。 在目标点之前的某个距离开始执行轨迹逼近 此距离最大可为起始点至目标点距离的一半。如果在此处输入了一个更大数值，则此值将被忽略而采用最大值。
2	轴加速 数值以设备数据中给出的最大值为基准。 • 1 … 100 %
3	圆周的姿态引导 • 以基准为参照 • 以轨迹为参照
4	轴速。 数值以设备数据中给出的最大值为基准。 • 1 … 100 %
5	姿态引导 • 标准 • 手动 PTP • 恒定的姿态引导

8.5 编程提示

SLIN-SPL-SLIN 过渡

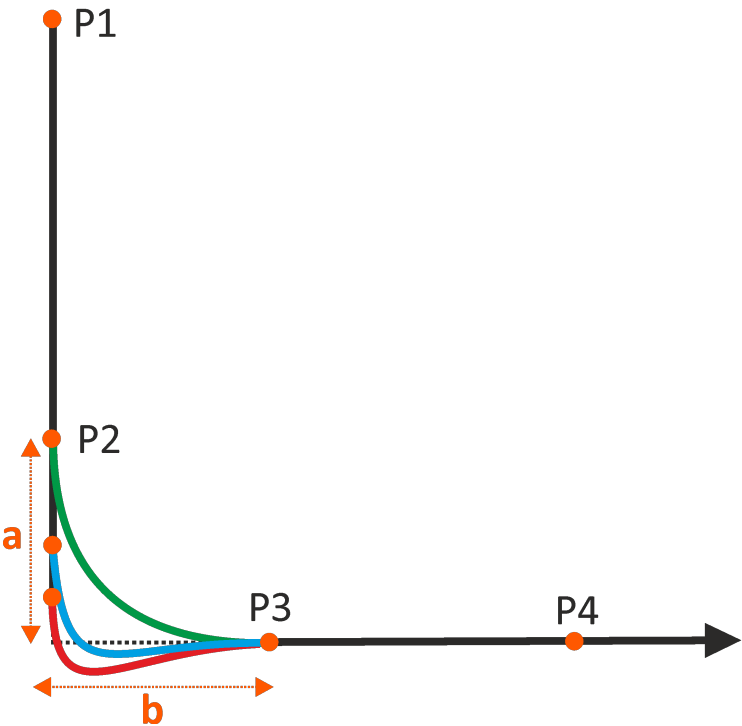


图 8-24: SLIN > SPL > SLIN 过渡

为避免在编程 SLIN > SPL > SLIN 过渡段时发生碰撞，必须注意以下几点：

- 两个 SLIN 段在其延长线上相交。
- $\frac{2}{3} \leq \frac{a}{b} \leq \frac{3}{2}$
a = 从 SPL 段的起点至 SLIN 段的交点的距离
a = 从 SLIN 段的交点至 SPL 段的目标点的距离

8.5.1 在样条组中更改点

说明

- **更改点的位置：**
如果移动了一个样条组中的一个点，则轨迹最多会在此点前的两个段中
和在此点后的两个段中发生变化。
小幅度的点平移通常不会引起轨迹变化。但如果相邻的两个段，一段非常
长而另一段非常短，则小小的变化就会产生非常大的影响。
- **更改段类型：**
如果将一个 SPL 段变成一个 SLIN 段或反过来，则前一个段和后一个
段的轨迹会改变。

示例 1

- **原有轨迹：**

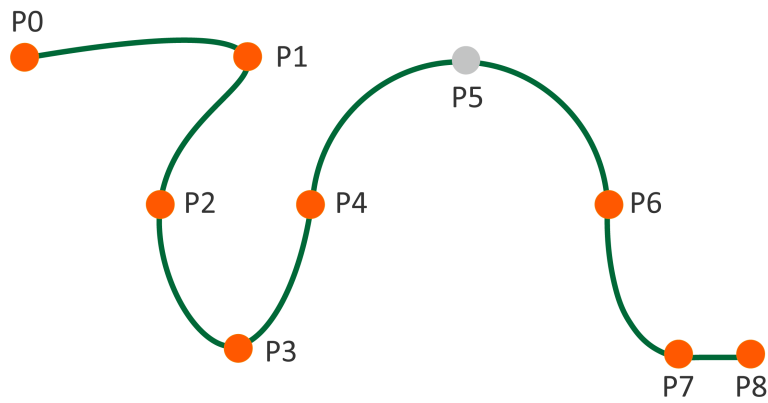


图 8-25: SPLINE, 原有轨迹

```
SPTP P0
SPLINE S1
  SPL P1
  SPL P2
  SPL P3
  SPL P4
  SCIRC P5, P6
  SPL P7
  SLIN P8
ENDSPLINE
```

- 与原有轨迹相比，点 3 发生移动：

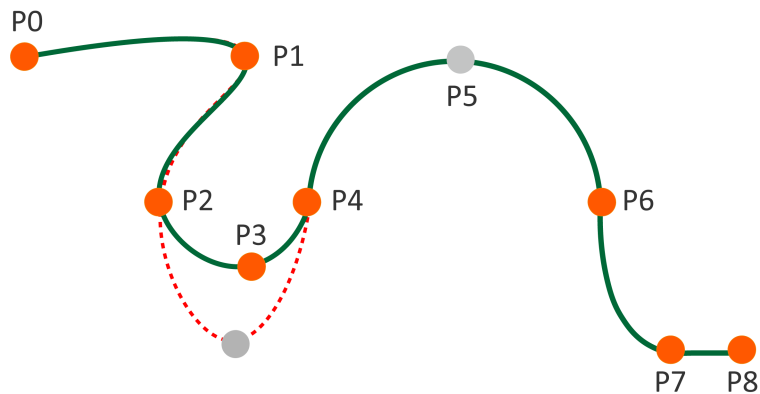


图 8-26: 点被移动

- P3 被移动。
 - 由此会改变 P1 - P2、P2 - P3 和 P3 - P4 段的轨迹。
 - 在这种情况下，P4 - P5 段不改变，因为它属于 SCIRC 并由此确定圆周轨迹。
- 更改原有轨迹的轨迹类型：

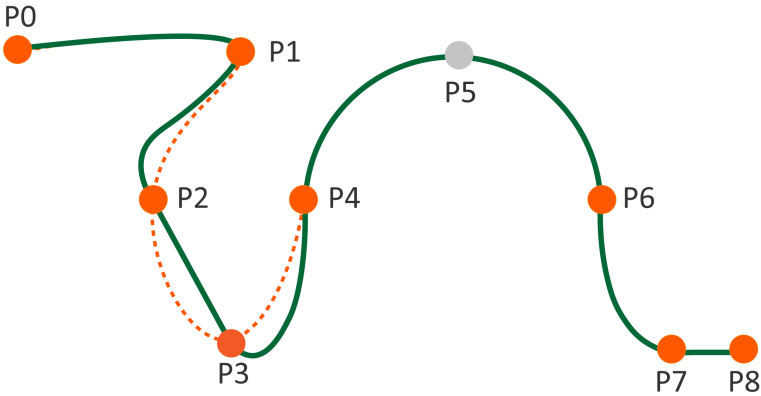


图 8-27：轨迹类型已更改

```
SPTP P0
SPLINE S1
  SPL P1
  SPL P2
  SLIN P3
  SPL P4
  SCIRC P5, P6
  SPL P7
  SLIN P8
ENDSPLINE
```

- 原来轨迹的 P2 - P3 段类型由 SPL 变为 SLIN。
- P1 - P2、P2 - P3 和 P3 - P4 段的轨迹发生变化。

示例 2

在该示例中一段行程全部用 SPL 段编程。

- 原有轨迹：

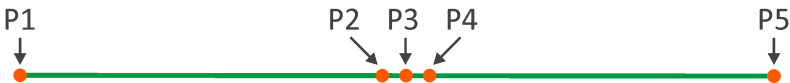


图 8-28：SPL，原有轨迹

- 移动原有轨迹的一个点：

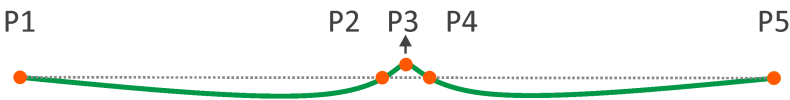


图 8-29：SPL，点平移

- P3 被移动。
- 由此会改变所有图示段中的轨迹。
- 因为 P2 - P3 段和 P3 - P4 段很短，而 P1 - P2 段和 P4 - P5 段很长，所以很小的移动也会使轨迹发生很大的变化。



- 均匀分配点的间距
- 将直线（除了很短的直线）作为 SLIN 段编程

8.5.2 样条曲线的速度说明

样条速度

- 轨迹曲线始终保持不变，不受倍率、速度或加速度的影响。
- 机器人控制器在规划时就已经考虑到机器人的物理极限了。
- 与原来的 LIN 和 CIRC 运动相比，这是一个优点，因为在这两种运动中不会在规划时考虑物理极限。
- 物理极限在运动期间才会起作用并在必要时触发停止。

减速

必须低于编程设定速度的情况主要包括：

- 突出的角
- 姿态改变过大
- 附加轴进行较大运动
- 在奇点附近



通过选择姿态引导 **无取向**，可在样条段中避免因姿态改变过大而引起的减速。

速度减至 0

以下情况：

- 连续的点具有相同坐标
- 样条组内有连续的 SLIN 段和/或 SCIRC 段。

原因：速度方向的间断变化。

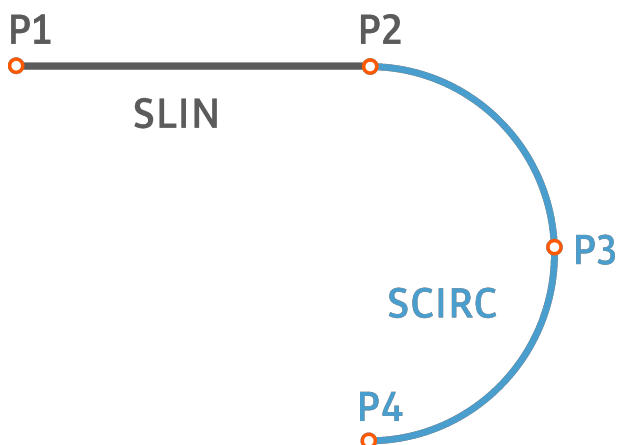


图 8-30：在 P2 点处精确暂停

在 SLIN 与 CIRC 的过渡段，当直线与圆相切时，由于圆与直线不同是曲线，因此速度也降为 0。

- 样条组内有 SLIN 段接着 SLIN 段。

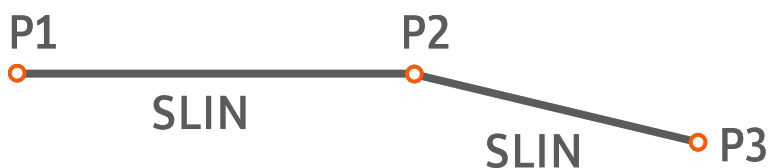


图 8-31：在 P2 点处精确暂停

补救措施

通过在 SLIN 和 SCIRC 之间插入一个 SPL 段，可避免精确停止，机器人可以编程设置的速度在工件上运动。

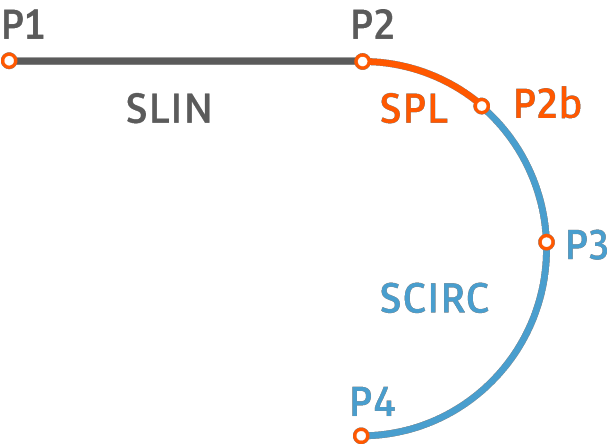


图 8-32：通过 SPL 在 SLIN-SCIRC 间过渡

例外

- 如果 SLIN 段是连续的，并构成一条直线且姿态均匀变化，则速度不会降低。



图 8-33：在 P2 点处不暂停而直接前行

- 对于 SCIRC 和 SCIRC 的过渡，如果两个圆或圆弧的圆心和半径一样，并且姿态均匀变化，则速度不会降低。（很难进行示教，所以需要点对点进行计算和编程）



有时对圆心和半径相同的圆进行编程，以保证圆 $\geq 360^\circ$ 。一个更简单的方法是对圆心角进行编程。

由于不均匀的示教造成的减速

- 当姿态变化（包括与附加轴有关）不能均匀分布至笛卡尔弧长时，通常会造成意料之外的速度急降。
- 姿态距离不均匀地分配给笛卡尔距离（弧长）时，姿态必须经常加速或者减速，容易伴随较大的姿态加速度变化率。
- 因此，距离分配不均匀时发生速度急降的频率比姿态距离分配均匀时高得多。

补救措施



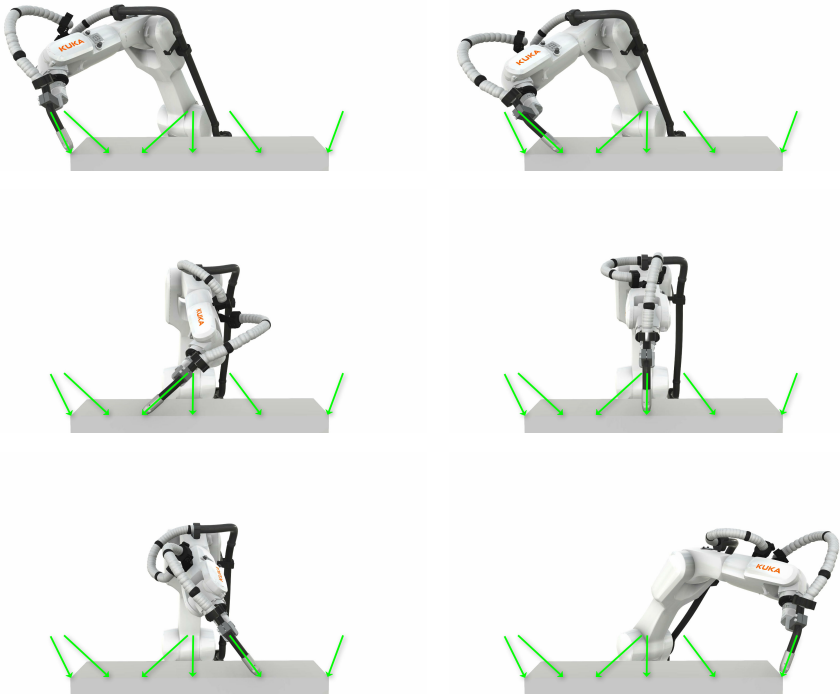
定向和附加轴要尽量均匀地分布

均匀分布的示例：

```
; 样条的起点
SPTP {x 0, y 0, z 0, A 0, B 0, C 0}
SPLINE S1
; 0,1° 改变姿态，每 mm 笛卡尔位移
SPL {x 0, y 100, z 0, A 10, B 0, C 0}
; 0,1° 改变姿态，每 mm 笛卡尔位移
SPL {x 0, y 110, z 0, A 11, B 0, C 0}
```

```
; 0,1° 改变姿态，每 mm 笛卡尔位移  
SPL {x 0, y 310, z 0, A 31, B 0, C 0}  
ENDSPLINE
```

带编程姿态



绿色箭头

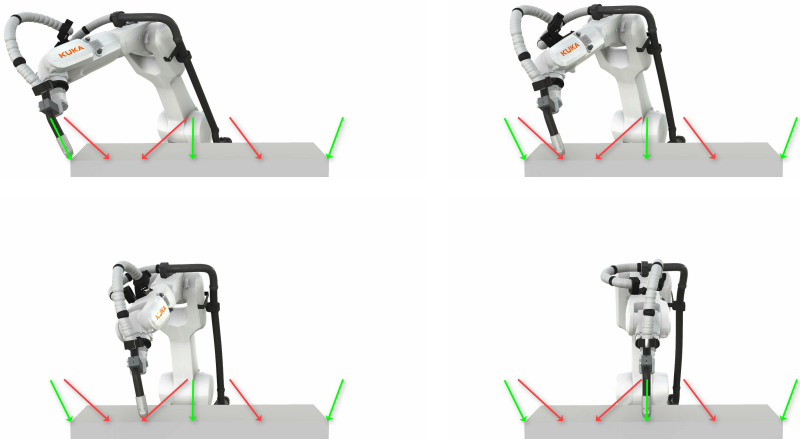
带相应姿态的示教位置

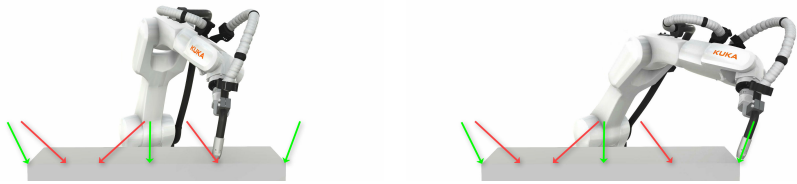
结论

提示

- 下面的情况经常发生：很多需要编程的点距离很近，
- 我们只是需要使用笛卡尔位置 (x, y, z)。
- 但样条组还是会考虑编程的姿态，这样会导致速度下降。
- 因此，这种情况下应在行指令中优先选择 **IGNORE**。

无编程姿态





绿色箭头
红色箭头

带相应姿态的示教位置
示教位置带相应姿态，其姿态未应用。

8.5.3 样条运动的语句选择

样条组

可在样条组的段上选择语句。

- **CP 样条组：**
BCO 运行将作为常规 LIN 运动被执行。
- **PTP 样条组：**
BCO 运行将作为常规 PTP 运动被执行。

选择语句后，轨迹通常与正常程序运行中的样条完全一样。如果在选择语句前尚未运行样条，并且在这种情况下语句选择于样条组始端，则可能会出现例外：

样条运动的起始点是样条组前的最后一个点，即起始点位于样条组之外。机器人控制器在正常运行样条时保存该起始点。如果以后要选择语句，则由此已知该起始点。但如果从未运行过样条组，则起始点未知。

示例：在 P1 点选择语句时轨迹的改变

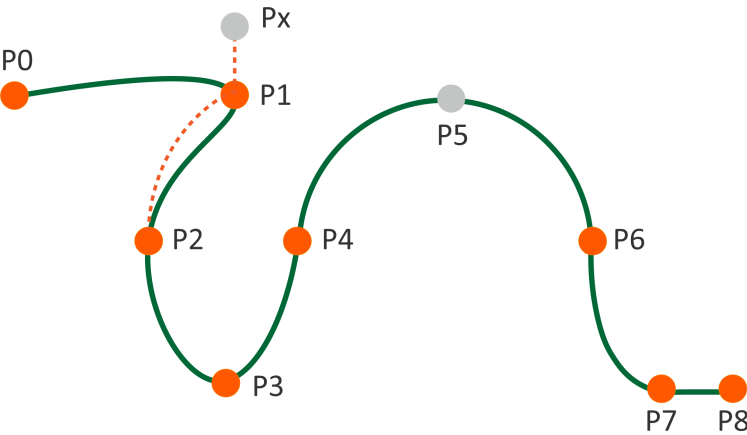


图 8-34：示例：在 P1 点选择语句时用未知 P0 改变的轨迹

行	说明
2	CP 样条组的标题行/起始
3 ... 9	样条段
10	CP 样条组的末尾

```
1 SPTP P0
2 SPLINE S1
3 SPL P1
4 SPL P2
5 SPL P3
```

```

6   SPL P4
7   SCIRC P5, P6
8   SPL P7
9   SLIN P8
10  ENDSPLINE

```

SCIRC

- 在针对一个编程设置的圆心角的 SCIRC 段语句选择时，机器人将移动到目标点（包括圆心角），前提条件是机器人控制系统可以识别起始点。
- 如果机器人控制系统无法识别起始点，则移向编程设定的目标点。在这种情况下会显示一则信息，提示未考虑圆心角。
- 在 SCIRC 单个运动上选择语句时，从不考虑圆心角。

8.6 样条曲线的轨迹逼近

说明

所有样条组和所有样条单个运动均可相互轨迹逼近。



样条运动无法用常规运动（LIN、CIRC、PTP）进行轨迹逼近。

因为时间或预进停止而无法轨迹逼近：

如果因为时间原因或因预进停止而无法轨迹逼近，则机器人在轨迹逼近弧线的起始处等待。

- 如果因为时间原因：只要可以规划下一个语句，则机器人继续移动。
- 如果因为预进停止：轨迹逼近弧线的起始处即是当前语句的终点。当预进停止被取消，机器人控制器可以规划下一个语句。机器人继续移动。

在这两种情况下，机器人沿轨迹逼近弧线移动。确切地说可以轨迹逼近，只是时间上会有推迟。

8.7 练习：带样条组的轨迹轮廓

练习内容

打开课程练习本并完成以下练习：

- 带样条组的轨迹轮廓

9 利用机器人程序中的逻辑功能

9.1 学习单元：在机器人程序中使用逻辑函数

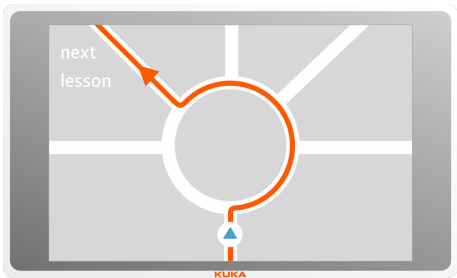


图 9-1：导航器

本学习单元讲授以下内容：

- 逻辑编程介绍
- 变量的显示
- 简单的逻辑编程
- 样条曲线的逻辑编程

9.2 逻辑编程介绍

说明

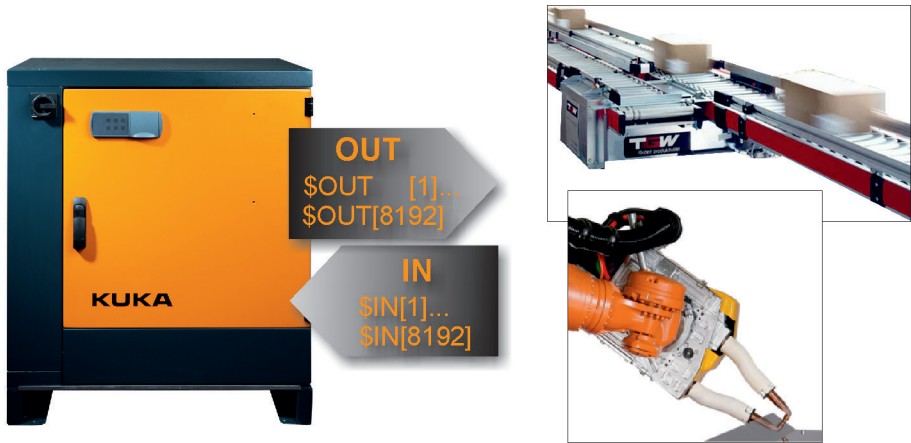


图 9-2：数字输入端和输出端

- 为了在定义的程序位置控制工具或等待信号等，控制系统可以使用现场总线接口。
- 该接口用于与外围设备进行通信，通过该外围设备可使用数字和模拟输入端和输出端。

概念解释

术语	说明	示例
通信	通过接口交换信号	查询状态（抓爪打开/闭合）
外围设备	“周围设备”	工具（例如抓手、焊钳等）、传感器、材料输送系统等等

术语	说明	示例
数字量	数字技术：数值和时间离散的信号	传感器信号：工件存在：值 1 (TRUE/真)，工件不存在：值 0 (FALSE/假)
模拟量	模拟一个物理量	温度测量
输入端	通过现场总线接口到达控制系统的信号	传感器信号：抓爪已打开/抓爪已闭合
输出端	通过现场总线接口从控制系统发送至外围设备的信号	用于闭合抓手的阀门输出指令。

9.3 显示变量

显示和更改一个变量的值

1. 菜单路径：机器人按键 > 显示 > 变量 > 单个



图 9-3：菜单路径

2. 单项变量显示窗口即打开。



图 9-4：窗口，单项变量显示

序号	说明
1	应显示或更改的变量的名称
2	该变量应被赋予的新值

序号	说明
3	在其中查找该变量的程序 对于系统变量来说， 模块 栏并不重要。
4	该栏位显示变量的值。有两种方法确保变量最新： ：显示的数值不自动更新。 ：显示的数值将自动更新。 在各状态间进行切换： - 按下更新。

3. 显示变量值

- 在**名称**栏输入变量名称。
- 如果选择了一个程序，则在**模块**栏中将自动填写该程序。
如果要显示一个其他程序中的变量，则如下输入程序：
/R1/ 程序名称
在 /R1/ 和程序名称之间不要输入文件夹。对程序名称不要输入文件扩展名。
- 按下回车键或“更新”按钮。
在**当前值**栏中将显示该变量的当前值。如果无任何显示，则说明还未给该变量赋值。
- 在**新值**栏中输入所需数值。
必须有足够的用户权限。
- 设定数值**按钮。
在**当前值**栏中将显示此新值。

9.3.1 显示输入端和输出端

说明

基于现场总线的数字和模拟输入端和输出端的当前状态可以显示在 smartPAD 上。可以通过手动输入设置输出端。
为此需要相应的用户权限。

操作步骤

- 菜单路径：** 机器人按键 > 显示 > 输入/输出端



图 9-5：菜单路径：输入端和输出端

- 切换到所需的显示 (1)。
 - 数字输入端和输出端
(>>> “数字输入端和输出端” 页面 260)
 - 模拟输入端和输出端

数字输入端和输出端

数字输出端示例

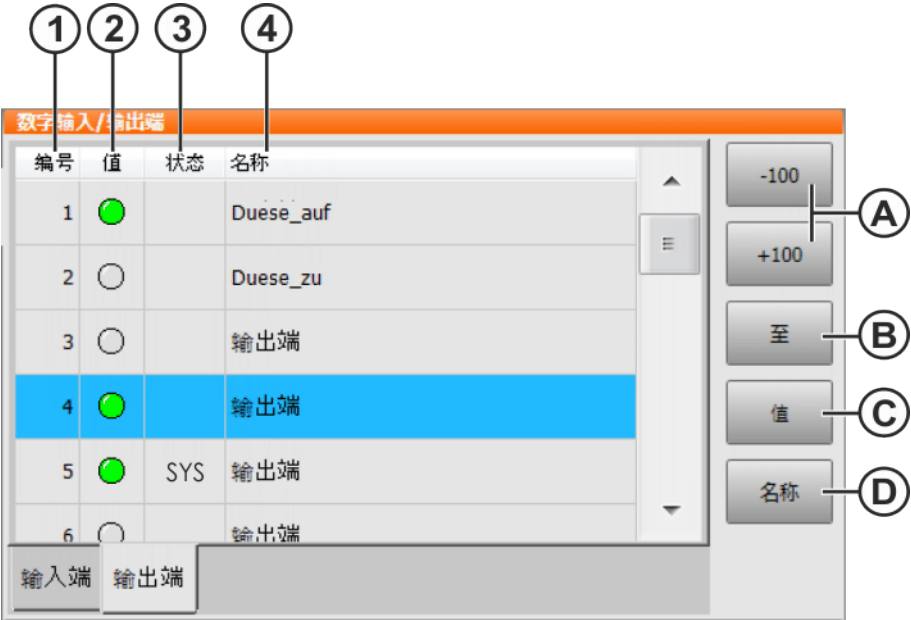


图 9-6：数字输出端

序号	说明	
1	输入/输出端编号	
2	输入/输出端数值。如果一个输入或输出端为 TRUE，则被标记为绿色。	
3	SIM 输入：已模拟输入/输出端。 SYS 输入：输入/输出端的值储存在系统变量中。此输入/输出端已写保护。	
4	输入/输出端名称 可更改已选中的输入/输出端的名称。	

编号	按键	说明
A	-100	在显示中切换到之前的 100 个输入或输出端。
	+100	在显示中切换到之后的 100 个输入或输出端。
B	至	可输入需搜索的输入或输出端的编号。
C	值	将选中的输入或输出端在 TRUE 和 FALSE 之间转换。前提条件：使能开关已按下。 • 在 AUT EXT（外部自动运行）运行模式下，值按键不可用。 • 仅当模拟已接通时，值按键才可用于输入端。

编号	按键	说明
D	名称	选中的输入/输出端的名称可更改。

9.3.2 显示旗标、计数器和计时器

- 菜单路径：机器人按键 > 显示 > 变量



图 9-7：菜单路径

这时便有不同的系统变量可供选择：

- 周期性旗标 (CycFlag)
- 标识器 (Flag)
- 计数器
- 定时器
- 示例：计数器

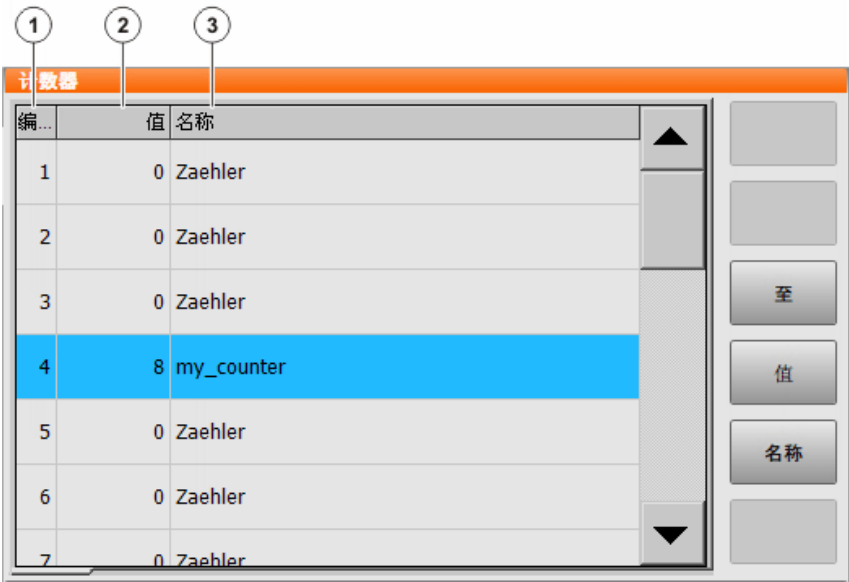


图 9-8：计数器

序号	说明
1	计数器的编号
4	计数器的值
5	计数器的名称

按键	说明
至	可输入需搜索的计数器编号。
值	可为选定的计数器输入一个数值。 (具备足够的用户权限)
名称	可更改选定的计数器名称。

9.4 简单的逻辑编程

说明

对 KUKA 机器人编程时，使用的是表示逻辑指令的输入端和输出端信号。

- **OUT**
在程序中的某个点上控制输出端
- **WAIT FOR**
取决于信号的等待功能 » 控制系统在此等待信号：
 - 输入端 **IN**
 - 输出端 **OUT**
 - 时间信号 **TIMER**
 - 控制系统内部的存储地址（标志位/1 bit） **FLAG** 或者 **CYCFLAG**（如果需要循环更新）
- **WAIT**
取决于时间的等待功能 » 控制系统根据输入的时间在程序中的点上等待。



在此所述的逻辑命令也是早期软件版本的基本组件。使用该功能始终是精确切换点和预进停止之间的折中方法。
建议在创建新项目时结合 SPTP、SLIN、SCIRC 和样条组检查触发指令（SYNOUT，在轨迹上切换）的使用情况。
在此可确保精确的切换点，这不取决于预进停止。

9.4.1 等待功能的编程

说明

计算机预进

```
1 DEF Depal_Box1()  
2  
3 INI  
4 SPTP HOME VEL= 100 % DEFAULT  
5 SPTP P1 VEL= 100 % PDAT1 Tool[5] Base[10]  
6 SPTP P2 VEL= 100 % PDAT1 Tool[5] Base[10]  
7 SLIN P3 VEL= 1 m/s CPDAT1 Tool[5] Base[10]  
8 SLIN P4 VEL= 1 m/s CPDAT2 Tool[5] Base[10]  
9 SPTP P5 VEL= 100% PDAT1 Tool[5] Base[10]  
10 OUT 26'' State=TRUE  
11 SPTP HOME VEL= 100 % DEFAULT  
12  
13 END
```

行	
6	主运行指针位置
9	可能的预进指针位置（不可见）
10	触发预进停止的指令语句

- 计算机预进时预先读入（操作人员不可见）运动语句，以便控制系统能够在有轨迹逼近指令时进行轨迹设计。
- 但处理的不仅仅是预进运动数据，而且还有数学的和控制外围设备的指令。
- 有些指令可触发预进停止。其中包括影响外围设备的指令，如 OUT 指令（抓爪关闭，焊钳打开）。
- 如果预进指针暂停，则不能进行轨迹逼近。

等待函数 WAIT

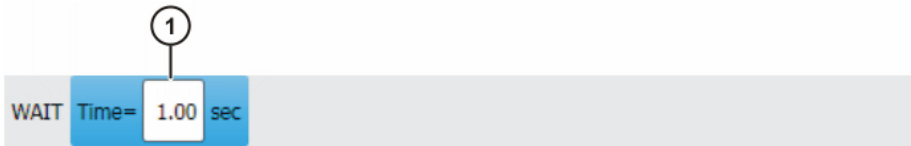


图 9-9：WAIT 的联机表格

序号	说明
1	等待时间 • $\geq 0\text{ s}$

- 运动程序中的等待功能可以很简单地通过行指令进行编程。
- 在这种情况下，等待函数被区分为与时间相关的等待函数和与信号相关的等待函数。
- 用 WAIT 可以使机器人的运动按编程设定的时间暂停。WAIT 总是触发一次预运行停止。

```
SPTP P1 Vel=100% PDAT1 Tool[1] Base[1]
SPTP P2 Vel=100% PDAT2 Tool[1] Base[1]
WAIT Time=2 sec
SPTP P3 Vel=100% PDAT3 Tool[1] Base[1]
```

示例

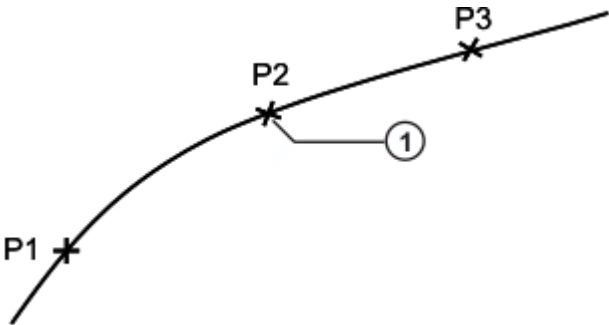


图 9-10：逻辑运动示例

序号	备注
1	在点 P2 处中断运动 2 秒钟。

等待功能 WAIT FOR

- 需要时可最多使用 12 个逻辑运算。

- 如果添加了一个逻辑运算，则行指令中会出现用于附加信号和其它逻辑运算的栏。

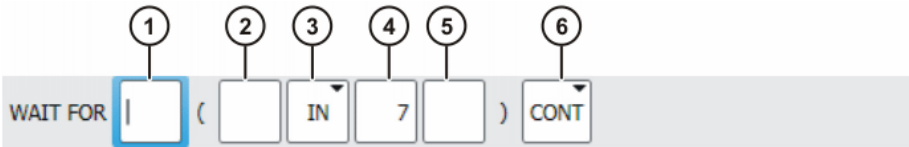


图 9-11：行指令 WAIT FOR

序号	说明
1	添加括号外部逻辑运算。运算符位于两个括号的表达式之间。 - AND - OR - EXOR 添加 NOT。 - NOT [空] 用相应的按键插入所需的运算符。
2	添加括号内部逻辑运算。运算符位于一个括号的表达式内。 - AND - OR - EXOR 添加 NOT。 - NOT [空] 用相应的按键插入所需的运算符。
3	等待的信号 - IN - OUT - CYCFLAG - TIMER - FLAG
4	信号的编号 1 ... 4096
5	如果信号已有注释则会显示出来。 仅限于专家用户组使用： 通过点击长文本可输入名称。名称可以自由选择。
6	- CONT：指令被预进指针执行 [空]：指令触发预进停止

逻辑运算

在应用与信号相关的等待功能时也会用到逻辑运算。
用逻辑运算可将不同信号或状态组合起来。

- 可定义相关性。

- 可以排除某些特定的状态。

一个具有逻辑运算符的函数始终以一个真值为结果，即最后始终给出“真”（值 1）或“假”（值 0）。

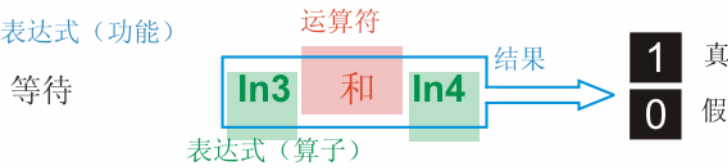


图 9-12：逻辑连接的范例和原理

逻辑运算的运算符为：

NOT	该运算符用于否定，即使值取反（由“真”变为“假”）。
AND	当连接的两个表达式为真时，该表达式的结果为真。
OR	当连接的两个表达式中至少一个为真时，该表达式的结果为真。
EXOR	当由该运算符连接的两个命题有不同的真值时，该表达式的结果为真。

与信号有关的等待功能在有预进或者没有其预进的处理下都可以进行编程设定。

不带预进的加工

没有预进的信号相关等待函数表示，在任何情况下都会将运动停在某点，并在该处检测信号：(1) (>>> 图 9-13) 该点不能轨迹逼近。

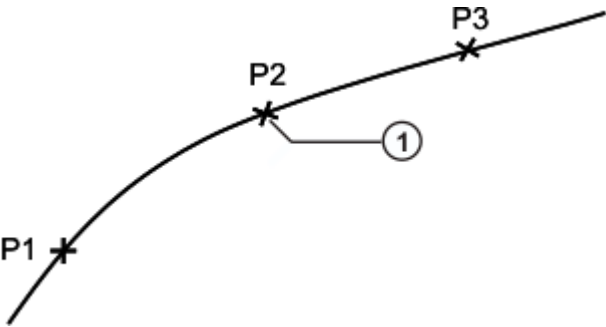


图 9-13：逻辑运动示例

```
SPTP P1 Vel=100% PDAT1 Tool[1] Base[1]
SPTP P2 CONT Vel=100% PDAT2 Tool[1] Base[1]
WAIT FOR IN 10 'door_signal'
SPTP P3 Vel=100% PDAT3 Tool[1] Base[1]
```



在执行无 CONT 的 WAIT FOR 行时，会显示一条提示信息：“无法轨迹逼近”。

有预进时的加工 (CONT)

- 有预进编程设定的与信号有关的等待功能允许在指令行前创建的点进行轨迹逼近。
- 但预进指针的当前位置却不唯一（标准值：三个运动语句），因此无法明确确定信号检测的准确时间 (1)
- 除此之外，信号检测后也不能识别信号更改！

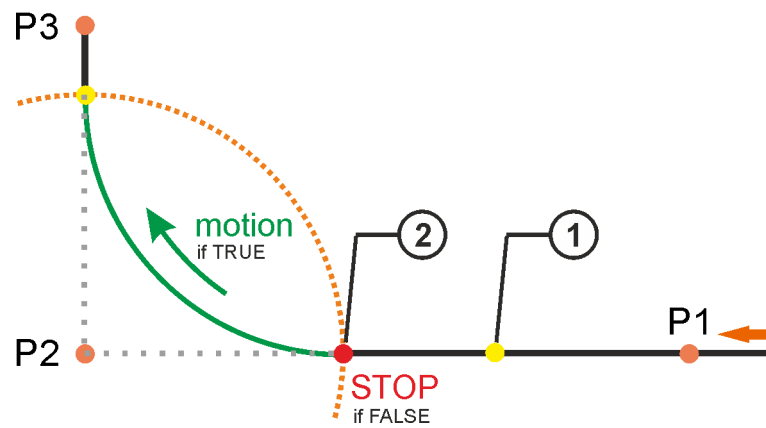


图 9-14: 轨迹逼近运动时的 WAIT FOR

```

SPTP P1 Vel=100% PDAT1 Tool[1] Base[1]
SLIN P2 CONT Vel=100% PDAT2 Tool[1] Base[1]
WAIT FOR IN 10 'door_signal' CONT
SLIN P3 Vel=100% PDAT3 Tool[1] Base[1]

```

1. 位置 (1)

- **WAIT FOR = TRUE** 在达到位置 (2) 之前
- » 机器人可以在不带精确暂停的情况下驶离轨迹逼近轨迹。

2. 位置 (2)

- **WAIT FOR = FALSE** 直达到达位置 (2)
- » 在带精确暂停的情况下机器人停留在轨迹逼近轮廓的开始位置。
- 在达到位置 (2) 后改为 **WAIT FOR = TRUE**
- » 机器人继续运行轨迹逼近轨迹上的轮廓。

操作步骤



图 9-15：插入逻辑

1. 将光标放到其后应插入逻辑指令的一行上 (1)。
2. 打开**命令**选项卡 (2)。
3. 通过**逻辑**条目 (3) 选择所需的等待或输出命令 (4)。
4. 在程序的标记位置上插入所需的行指令。
5. 在行指令中设置参数。
6. 用**命令 OK** 保存指令。

9.4.2 单一信号输出功能编程

说明

单一信号输出功能

- 通过输出功能可将数字信号传送给外围设备。

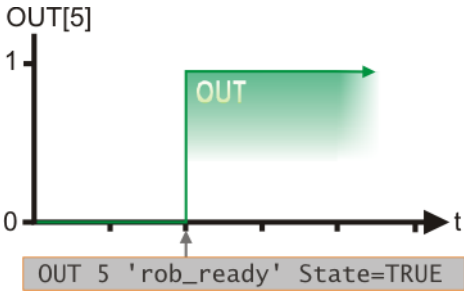


图 9-16：静态切换

- 为此要使用先前分配给接口的输出端编号。
- 信号为静态，即它一直存在，直至赋予输出端另一个值。
- 输出功能在程序中通过行指令实现：

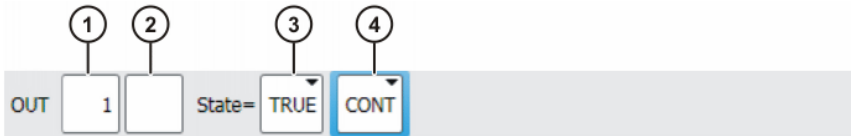


图 9-17：联机表格 OUT

序号	说明
1	输出端编号 – 1 ... 4096
2	如果输出端已有名称则会显示出来。 仅限于专家用户组使用： 通过点击长文本可输入名称。名称可以自由选择。
3	输出端被输出成的状态 – TRUE – FALSE
4	– CONT: 指令被预进指针执行 – [空]: 指令触发预进停止



使用简单的切换命令 OUT，在程序开始时将现场总线输出端切换为定义的输出端状态。

脉冲切换函数

- 与简单的切换函数一样，在此输出端的数值也变化。
- 然而，在脉冲时，于定义的时间过去之后，信号又重新取消。

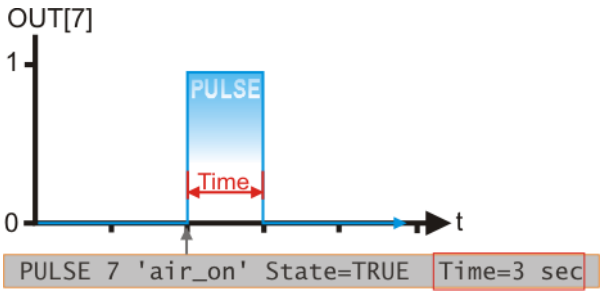


图 9-18：脉冲电平

- 编程同样使用行指令，在该行指令中给脉冲设置了一定的时间长度。

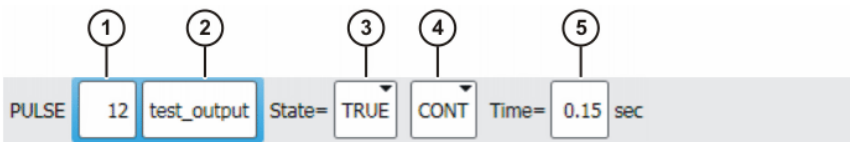


图 9-19：PULSE 的联机表格

序号	说明
1	输出端编号 – 1 … 4096
2	如果输出端已有名称则会显示出来。 仅限于专家用户组使用： 通过点击 长文本 可输入名称。名称可以自由选择。
3	输出端被切换成的状态 – TRUE: “高” 电平 – FALSE: “低” 电平
4	– CONT: 在预进过程中加工 – [空]: 带预进停止的加工
5	脉冲长度 – 0.10 …3.00 s

切换功能时 CONT 的影响

- 无 CONT:
如果在 OUT 行指令中去掉条目 CONT，则在切换过程时必须执行**预进停止**，并接着在切换指令前于点上进行精确暂停。给输出端赋值后继续该运动。

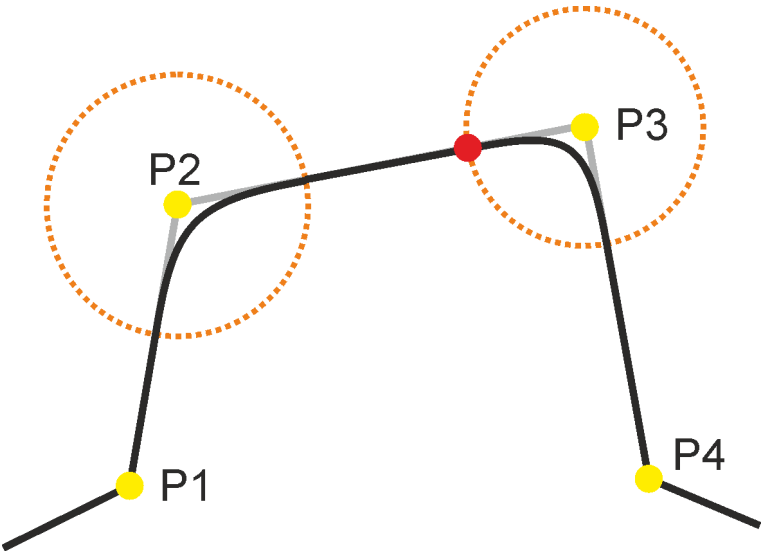


图 9-20：含切换和预进停止的运动举例

```
SLIN P1 Vel=0.2 m/s CPDAT1 Tool[1] Base[1]  
SLIN P2 CONT Vel=0.2 m/s CPDAT2 Tool[1] Base[1]  
SLIN P3 CONT Vel=0.2 m/s CPDAT3 Tool[1] Base[1]  
OUT 5 'rob_ready' State=TRUE  
SLIN P4 Vel=0.2 m/s CPDAT4 Tool[1] Base[1]
```

- 带 CONT:

插入条目 CONT 的作用是，预进指针不被暂停（不触发预进停止）。因此，在切换指令前运动可以轨迹逼近。在**预进**时发出信号。

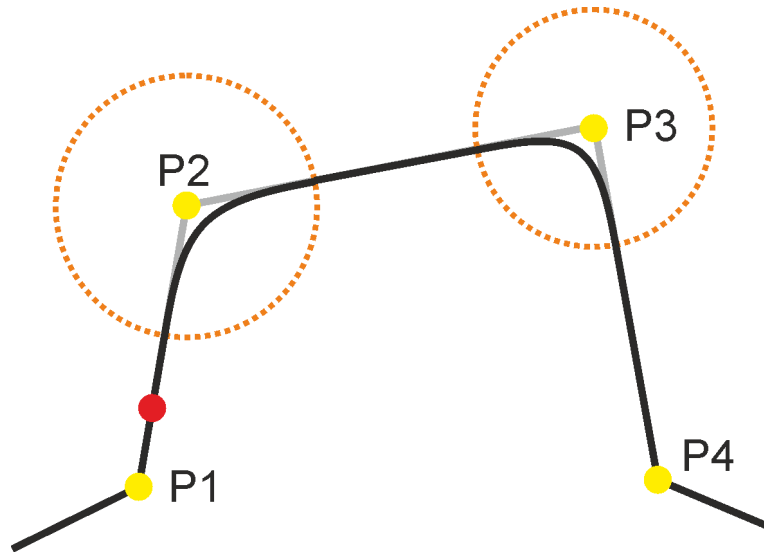


图 9-21：含切换和预进的运动示例

```
SLIN P1 Vel=0.2 m/s CPDAT1 Tool[1] Base[1]
SLIN P2 CONT Vel=0.2 m/s CPDAT2 Tool[1] Base[1]
SLIN P3 CONT Vel=0.2 m/s CPDAT3 Tool[1] Base[1]
OUT 5 'rob_ready' State=TRUE CONT
SLIN P4 Vel=0.2 m/s CPDAT4 Tool[1] Base[1]
```



预进指针的标准值占三个运动语句。但预进是会变化的，即必须考虑到，切换时间点不是保持不变的！

操作步骤

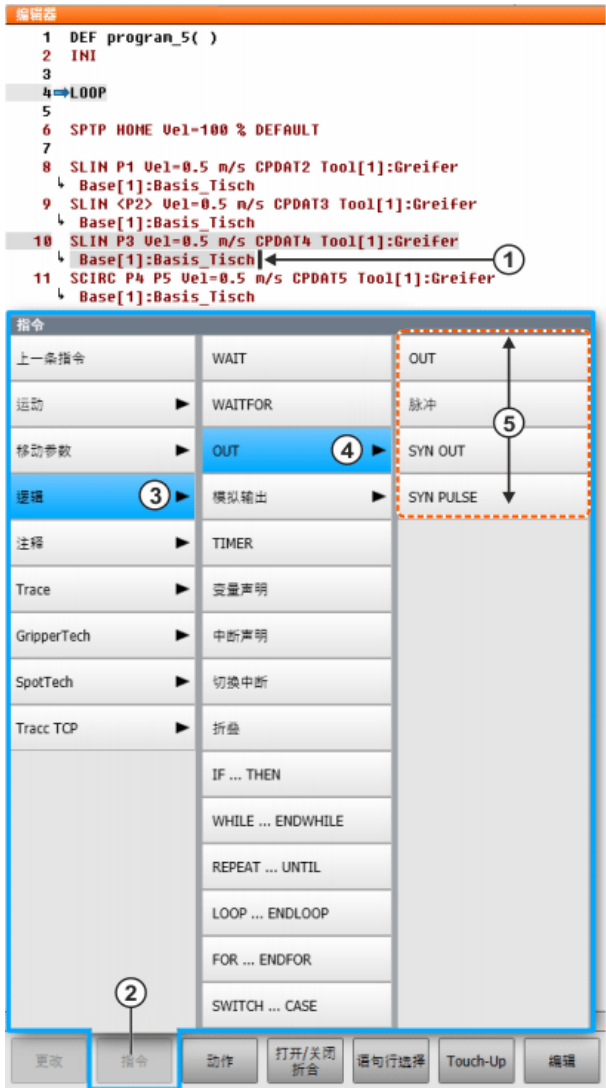


图 9-22：插入切换指令

- 1. 将光标放到其后需插入逻辑指令 (1) 的一行上。
- 2. 菜单序列：指令 (2) > 逻辑 (3) > OUT (4) > OUT、PULSE、SYN OUT 或 SYN PULSE (5)
- 3. 在行指令中设置参数。
- 4. 用指令 OK 保存指令。

9.4.3 可选练习：给轨迹运动配设逻辑

练习内容

打开课程练习本并完成以下练习：

- 给轨迹运动配设逻辑

9.5 编程带样条和样条单个语句的逻辑

说明

样条单个语句（SLIN、SCIRC、SPTP）和样条组编程时，除了集成运动外，还可以集成逻辑。可以通过 KRL 或更简便地通过行指令对逻辑编程。



图 9-23：行指令中的样条逻辑

始终可用（ADAT）：

- 触发器（轨迹上的输出指令）
- 条件停止

仅在样条组中

- 恒速运动区域
- 触发器作为行指令

仅在样条组中并且仅作为 KRL 指令

- 时间段

仅在样条组之外

- 条件停止作为行指令（Spline Stop Condition）

9.5.1 为样条触发器编程

说明

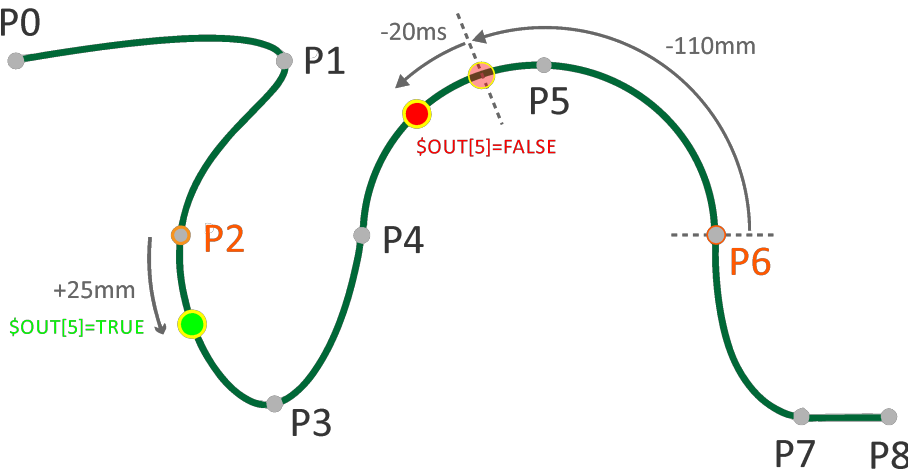


图 9-24：带样条的触发器

示例显示借助样条组编程的轨迹。触发器功能也可传输到轨迹运动上，该运动由单个语句组成。

触发器可以触发一个由用户定义的指令，示例为数字输出端 5。机器人控制器与机器人运动同时执行该指令。触发器可选择性地与运动的起始点或目标点关联起来。指令直接在参考点上触发或还可在位置和/或时间上偏移。



如果触发器用于样条组，则其不得位于上一段和 ENDSPLINE 之间。

编程方法

- 指令行选项窗口逻辑触发器
- 样条触发器指令行
- 使用 KRL 指令 TRIGGER WHEN PATH 编程

9.5.1.1 样条曲线触发器编程

操作步骤

1. 用按键“打开/关闭折叠”打开样条组
2. 在点 5 上应接通数字输出端 \$OUT[5]。

```
4  SPLINE S1 Vel=0.5 m/s CPDAT 2 Tool[1]:pen Base[5] blue
5  SPL P4
6  SLIN P5 ADAT
7  SPL P6
8  SLIN P7
9  SPL P8
10 ENDSPLINE
```

3. 选择参数

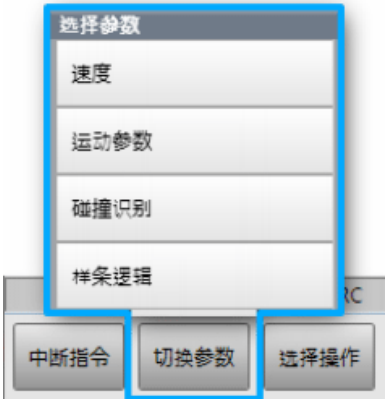


图 9-25：切换参数

- 给点 5 扩展样条逻辑。
 - 按钮：更改 > 切换参数 > 添加样条逻辑
4. 点 5

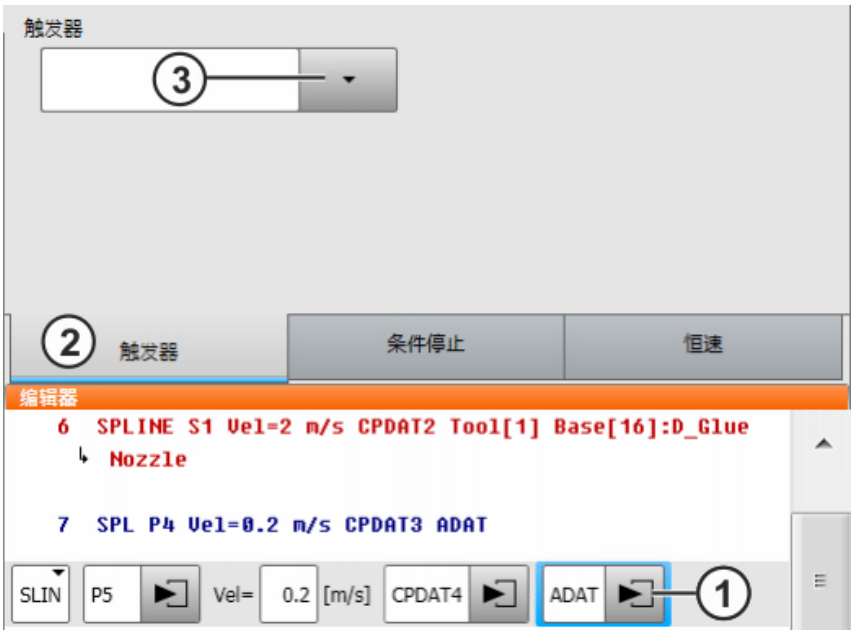


图 9-26：触发器选择（空）

- 点击点 5（ADAT）上的箭头符号（1）。
- 在显示的窗口中切换到触发器选项卡（2）中。



通过触发器下拉窗口可以从要在该点上激活的预定义触发器/切换动作中选择。这些触发器当前还不存在，必须在下一步中定义。

5. 添加触发器指令

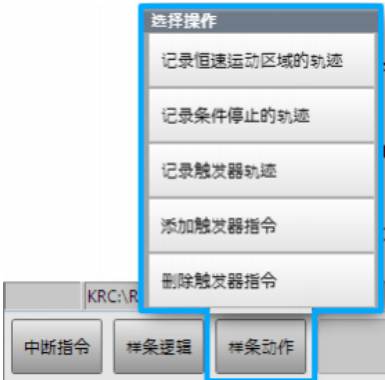


图 9-27：样条动作

- 在打开的运动语句中通过**样条动作**按键**添加触发命令**。
 - 通过其他的输入栏扩展触发器窗口。
6. **类型 1**：激活点上的触发器。



图 9-28：定义样条触发器

- **触发器 (1)**
系统创建一个名称为触发器 1 的新触发器。如果已经定义了其他的触发器，则可以通过这些窗口调用和调整这些触发器。
- **ONSTART (2)**
如果参数 ONSTART 被设置为 **TRUE**，则触发器触发的参考点指的是起始点。这可以通过参数 Path 和 Delay 额外移动。在我们的示例中为数字输出端 5。
- **触发器类型 (3)**

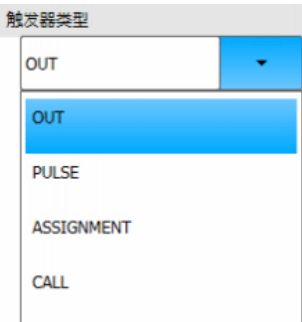


图 9-29：触发器类型

定义要使用哪种触发器。在此有时需要另外掌握 KRL 的专家知识。
在我们的示例中应选择 OUT，以便切换数字输出端。



取决于所选触发器的种类 PULSE、ASSIGNMENT 或 CALL，栏 3、4 和 7 中的输入窗口和选择窗口会发生变化。
详细信息请参阅软件文件。

- **OUT (4)**
选择通过该触发器切换哪个数字输出端。
- **PATH (5)**
因为触发器直接在点上切换并且为 TRUE，因此位置偏移未激活。
- **Delay (6)**
取决于运行速度，可以参考该点在时间上推迟该点。有助于例如涂胶
- **OUT 状态 (7)**

在此可以定义数字输出端的所需切换状态。在该示例中，通过布尔表达式 TRUE 接通。

- 7. 类型 2：让系统记录触发器值 Path（距离）。
在示例中：相对于点 5 的位置和时间偏移。

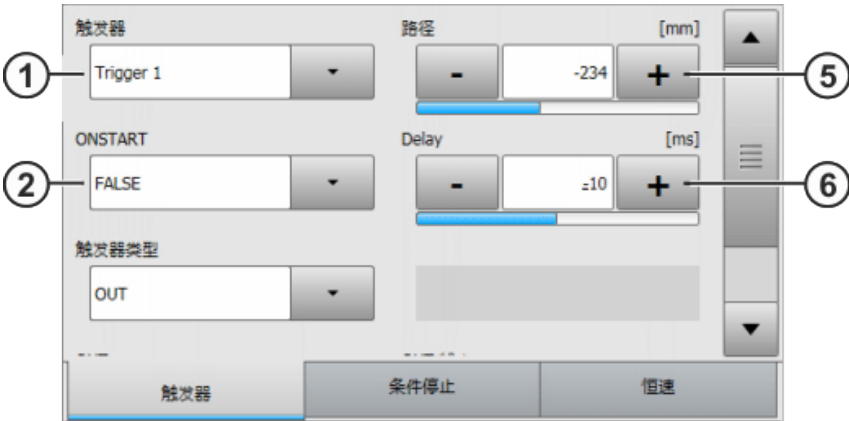


图 9-30：位置和时间偏移

- 在样条轨迹上借助程序键向前或向后移动机器人，直到到达“触发点”。
- 通过按键 样条动作 > 记录触发器 Path（轨迹）将当前位置应用到 Path 栏中。



图 9-31：样条动作



如果通过手动运行机器人离开该轨迹，则无法借助轨迹进行记录。此处需要重新选择语句。



通过示教将 ONSTART 始终设定为 FALSE 并指定相对于目标点（负）的位置偏移。



原则上，可以用 ONSTART 将相对于起点的偏移设定为 TRUE。但是，在此只能借助输入数字设定偏移量（正）。

- 在需要时，在 Delay [ms] 栏中用数字输入时间偏移。

9.5.1.2 借助行指令给样条曲线触发器编程

说明

除了直接在样条组的运动点上编程样条曲线触发器之外，还可以在一个单独的行指令中对其进行编程。



样条触发器行指令只能在样条组内使用。

操作步骤

- 1. 选中要在其后插入逻辑命令的运动语句。
- 2. 插入样条逻辑



图 9-32：插入样条逻辑

菜单路径：指令按钮 (1) > 运动参数 (2) > 样条触发器或样条停止条件 (3)

- 3. 配置显示的行指令。

行指令

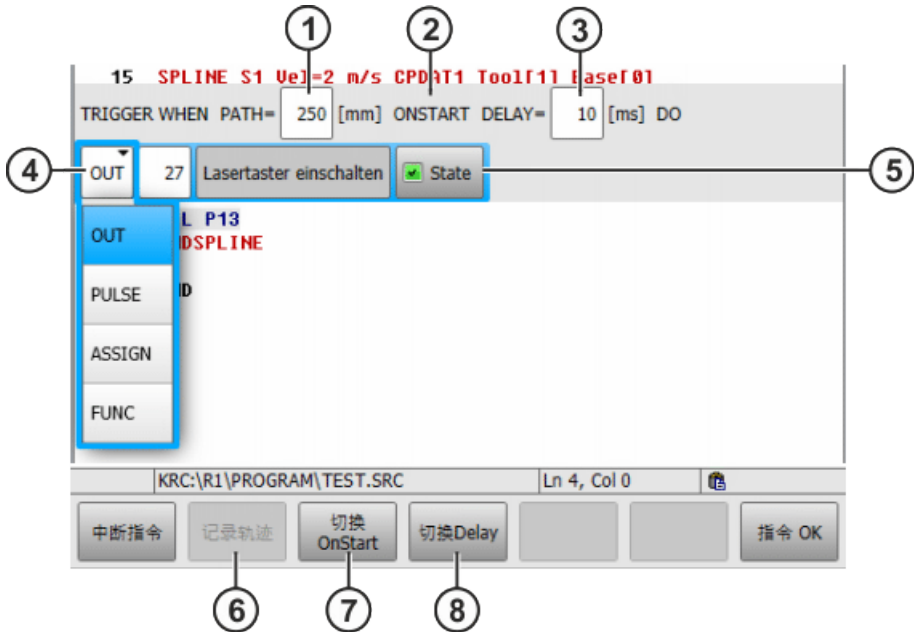


图 9-33：行指令，样条触发器

序号	名称	说明
1	记录轨迹	可以用记录 Path（轨迹）“示教”轨迹上切换点的位置偏移（Path）。
2	ONSTART	样条触发器参照的点 - 有 ONSTART：样条触发器前的最后一个点 - 无 ONSTART：样条触发器后的点 ONSTART 可通过按钮 切换 OnStart 加以设置或删除。
3	DELAY	样条触发器的时间偏移
4	OUT	触发器的触发反应 - OUT（设定输出端） - PUSLE（设定脉冲输出端） - ASSIGN（触发器分配） - FUNC（触发器函数调用）
5	State（状态）	指定待采用的输出端状态 TRUE = 按钮打绿勾 FALSE = 按钮显示灰色
6	记录轨迹	通过驶向轨迹上的触发点自动确定 PATH 值
7	切换 OnStart	参见项号 2
8	切换 Delay	参见项号 3

示例：调出一个子程序

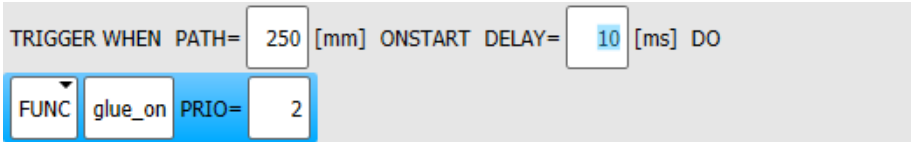


图 9-34：中断示例

9.5.1.3 编程提示

问题

轨迹逼近起始点或者目标点时，PATH 触发器的参考点在哪里？



该参考点主要取决于涉及均匀轨迹逼近或混合轨迹逼近。

均匀轨迹逼近

- 从 CP 样条运动进入 CP 样条运动
- 从 PTP 样条运动进入 PTP 样条运动
- 从 LIN 或 CIRC 运动进入 LIN 或 CIRC 运动



每个样条运动可能是样条组或单个语句。

混合轨迹逼近

- 从 CP 样条运动进入 PTP 样条运动，或反过来
每次样条运动可能是样条组或单个语句。
- 从 PTP 运动进入 LIN 或 CIRC 运动，或反过来



参考点的位置还取决于涉及样条运动还是常规运动。

专家文件

关于“轨迹相关的切换动作（触发器）”主题的详细信息请查阅专家文件。



Xpert 搜索: **KUKA System Software 8.6**
 面向系统集成商的操作及编程指南
 Xpert 筛选器: 文献 > 操作与编程指南
 章节: KRL 编程 > 轨迹相关的切换动作 (=触发器)

9.5.2 为条件停止编程

说明

“条件停止”允许用户定义在某个位置上满足特定条件时，机器人停止。该位置称为“停止点”。如果不再满足该条件，则机器人继续运行。机器人控制器在运行期间计算出最迟必须制动的点，以便能够在停止点停止。从该点（=“制动点”）起，机器人控制器分析是否满足条件。如果在制动点上满足条件，则机器人制动，以便在停止点停止。但如果在到达停止点前重新变为“不满足”条件，则机器人重新加速，而不会停止。如果只在机器人已经通过制动点时才满足停止条件，那就太迟了，无法以正常制动斜坡在停止点上停止。

- 在这种情况下，则机器人顺沿轨迹紧急停止，并停在不可预见的点上。
- 如果机器人在停止点后通过紧急制动停止，则程序在不再满足条件时才能继续运行。

如果机器人在停止点前通过紧急制动停止，则程序继续运行时将出现以下情形：

- **不再满足**停止条件时：
机器人继续移动。
- **仍满足**停止条件时：
机器人会移动至停止点，并在该处停止。

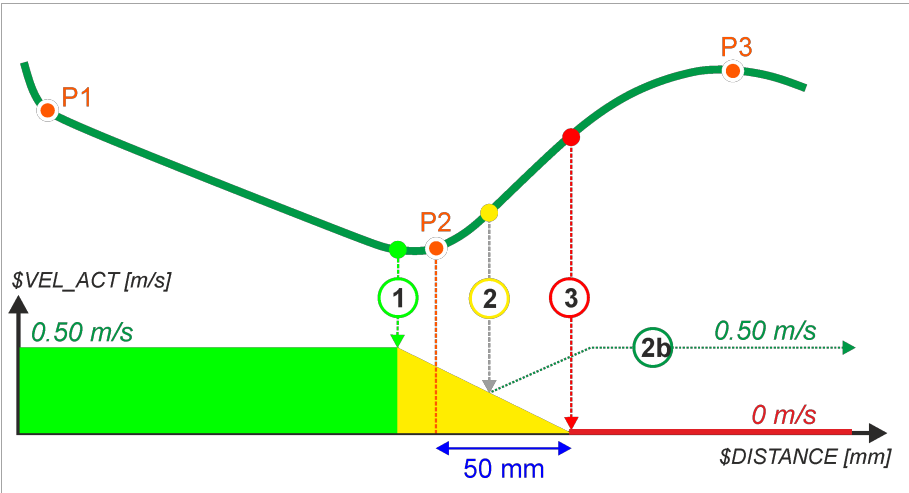


图 9-35：带条件停止的样条

序号	说明
1	计算所得的制动点，取决于行程和速度
2	制动过程中可能发生状态变化（不再满足条件）。机器人重新加速至编程设定的速度。
2 b	在重新加速之后达到编程设定的速度。
3	规定的停止点（通过行指令）



原则上可以任意编程设定多个条件停止。但最多允许有 10 段“制动点 > 停止点”相交。
在制动过程中，机器人控制器在 T1/T2 下显示以下信息：条件停止激活（行 {行号}）。

编程方法

- 在样条组或样条单个语句中：
指令行选项窗口逻辑条件停止
- 在样条组前：
以样条指令行为条件的停止

9.5.2.1 编程提示

检查数字输入端

\$IN[1]==TRUE	
\$IN	在此是一个由系统规定的栏（变量）。 投入运行人员将现场总线用户数字输入端的当前状态映射到该栏中。状态可能为 TRUE（接通）或 FALSE（关闭）。
[1]	该数字等于栏下标。在该情况下，数字输入端为编号 1。
==	比较运算；将输入端接线柱的实际物理状态（行 1）与该表达式右侧的状态（行 4）进行比较。
TRUE	预设值，数字输入端应为接通。如是，则触发条件停止。 如果要比较数字输入端是否为关闭状态，则比较运算符必须为 FALSE。



为了监控一个数字输入端是否为 TRUE，也可以使用以下缩短的输入。
\$IN[1]

9.5.2.2 条件停止编程

操作步骤

1. 用按键“打开/关闭折叠”打开样条组
2. 条件停止应在点 4 上触发。

```
4  SPLINE S1 Vel=0.5 m/s CPDAT 2 Tool[1]:pen Base[5] blue
5  SPL P1
6  SLIN P2
7  SPL P2
8  SLIN P4
9  SPL P5
10 ENDSPLINE
```

3. 插入样条逻辑

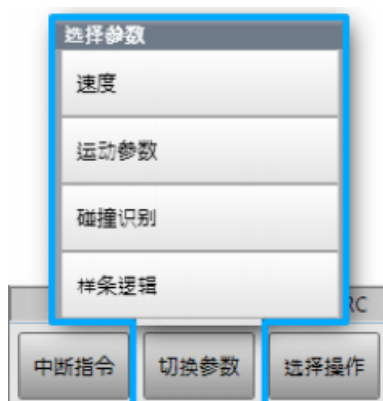


图 9-36：切换参数

- 给运动点 P4 扩展样条逻辑（ADAT）。
- 按钮：更改 > 切换参数 > 添加样条逻辑

```
4  SPLINE S1 Vel=0.5 m/s CPDAT 2 Tool[1]:pen Base[5] blue
5  SPL P1
6  SLIN P2
7  SPL P2
8  SLIN P4 ADAT
9  SPL P5
10 ENDSPLINE
```

4. 激活条件停止



图 9-37：激活条件停止

- 点击点 4 上的箭头符号 (1)。
 - 在显示的窗口中切换到条件停止选项卡 (2) 中。
 - 通过下拉窗口激活条件停止 (3)。
- 为此，选择 **ENABLED**。

5. 编程设置条件并输入手动偏移量。



图 9-38：为条件停止编程

序号	参数	值	说明
1	条件停止	ENABLE D	已参照该点激活了条件停止
2	条件	KRL	为了在驶向该运动点时触发条件 停止或不停止地继续运行要询问 的条件。 用 KRL 编程该条件。 (>>> “检查数字输入端” 页 面 281)
3	ONSTART	TRUE	启动指的是起点 如果起始点已被轨迹逼近，则可 以与 PATH 触发器时均匀轨迹 逼近相同的方式得出参考点。
4	Path	0 mm	恒速运行区域起点没有位置偏移

型号

参照点 4 的条件停止的起点有位置偏移。



图 9-39：参考点已移动

序号	参数	值	说明
1	Path	-234 mm	将条件停止的参考点确定为达到目标点前 234mm。
2	ONSTART	FALSE	启动指的是目标点 如果目标点已被轨迹逼近，则 Start 或 End 以轨迹逼近弧线的起点为参照。

- 在样条轨迹上借助程序键向前或向后移动机器人，直到达到了条件停止的参考点。
- 通过按键 **样条动作** > **记录条件停止 Path (轨迹)**将当前位置应用到 Path 栏中。



图 9-40：样条动作



如果通过手动运行机器人离开该轨迹，则无法借助轨迹进行记录。此处需要重新选择语句。



通过示教将 ONSTART 始终设定为 FALSE 并指定相对于目标点（负）的位置偏移。



原则上，可以用 ONSTART 将相对于起点的偏移设定为 TRUE。但是，在此只能借助输入数字设定偏移量（正）。

9.5.2.3 借助行指令编程条件停止

说明

除了直接在样条组的运动点上编程条件停止之外，还可以在一个单独的行指令中对其进行编程。



该指令行只允许在样条组之外使用。

操作步骤

- 1. 选中要在其后插入逻辑命令的运动语句。
- 2. 插入样条逻辑



图 9-41：插入样条逻辑

菜单路径：指令按钮 (1) > 运动参数 (2) > 样条触发器或样条停止条件 (3)

- 3. 配置显示的行指令。

行指令

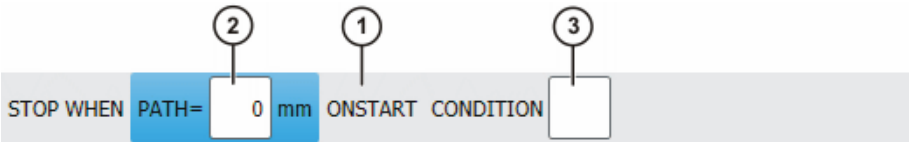


图 9-42：行指令 样条停止条件

序号	说明
1	条件停止参照的点 • 有 ONSTART：样条组前的最后一个点 • 无 ONSTART：样条组中的最后一个点 如已经轨迹逼近了样条，则适用与 PATH 触发器相同的规则。 提示：关于 PATH 触发器时轨迹逼近的信息，见系统集成商的操作/编程手册。 ONSTART 可通过按钮切换 OnStart 加以设置或删除。
2	可以移动停止点的位置。为此必须给出距参照点所要的距离。如果无需移动位置，则输入“0”。 • 正值：朝运动结束方向移动 • 负值：朝运动起始方向移动 停止点不可任意远移动位置。适用与 PATH 触发器相同的极限值。也可以示教位置移动。
3	停止条件。 允许使用： • 全局布尔变量 • 信号名称 • 比较 • 简单的逻辑联接：NOT、OR、AND 或 EXOR
导致生成错误信息：	
(\$IN[29]==FALSE) OR (\$IN[22]==TRUE)	
正确：	
NOT \$IN[29] OR \$IN[22]	

按钮

在行指令打开的状态下，还将显示待补充的按键。



图 9-43：按钮

序号	名称	说明
1	记录轨迹	可以用记录轨迹“示教”轨迹上切换点的位置偏移（Path）。
2	切换 OnStart	在无位置偏移的点上规定切换点

9.5.3 恒速运动区域

说明

如可能，在 CP 样条组中可以定义机器人恒定保持设定速度的区域。区域被称为“恒速运动区域”。

- 每个样条组可以定义 1 个恒速运动区域。
- 匀速运动区域通过一个起始指令和一个终止指令加以定义。
- 该区域不可以超出样条组。
- 该区域可以任意小。

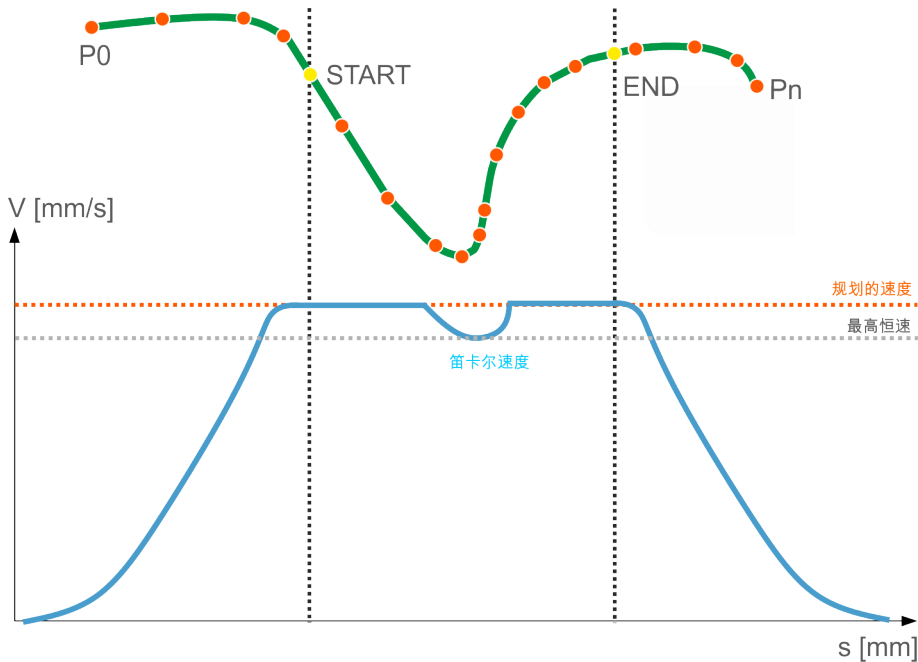


图 9-44：恒速，原理

如果无法恒定保持编程设定的速度，则机器人控制器会在测试模式内的程序运行时通过信息对此加以显示。

在多个段上的恒速运动区域：

- 一个恒速运动区域可以延伸至具有不同编程速度的多个段。
在这种情况下，整个区域适用编程设定的最低速度。
- 即使是在具有较高编程设定速度的段中，这时也以最低速度运行。
这时不会因为低于速度极限值而显示信息。
仅当不能保持编程设定的最低速度时才会显示信息。

9.5.3.1 恒速运动区域编程

操作步骤

1. 用按键“打开/关闭折叠”打开样条组
2. 在 P5 和 P12 之间应保持速度恒定。

```
4  SPLINE S1 Vel=0.5 m/s CPDAT 2 Tool[1]:pen Base[5] blue
5  SPL P4 CPDAT4
6  SLIN P5 ADAT
7  SPL P6
8  SLIN P7
9  SPL P8
10 SPL P9
11 SPL P10
12 SPL P11
13 SLIN P12ADAT
14 SPL P13
15 ENDSPLINE
```

3. 在点 5 上开始和在点 12 上结束恒速运行。

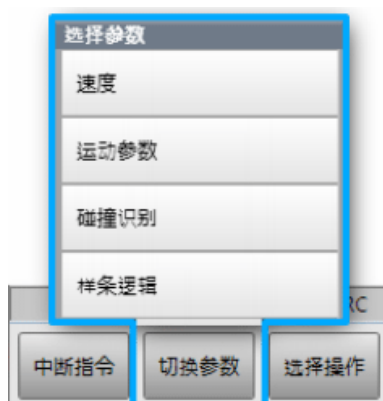


图 9-45：切换参数

- 即必须在这两个位置（P5 和 P12）上给运动点扩展样条逻辑（ADAT）。
- **按钮：**更改 > 切换参数 > 添加样条逻辑

4. 点 5



图 9-46：激活恒速

- 点击点 5 上的箭头符号（1）。
- 在显示的窗口中切换到恒速选项卡（2）中。
- 通过下拉窗口激活恒速（3）。
为此，选择 **START**。

5. 编程设置手动偏移量。

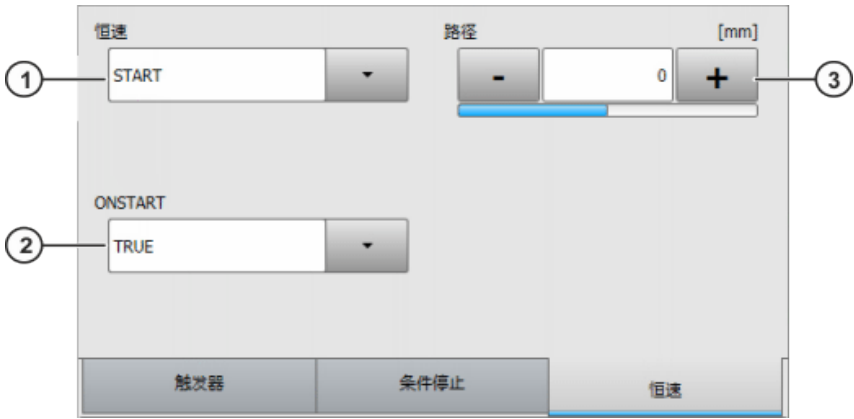


图 9-47：激活恒速

参数	值	说明
恒速	启动	从恒速运行区域起点开始
ONSTART	TRUE	启动指的是起点 如果起始点已被轨迹逼近，则可以以与 PATH 触发器时均匀轨迹逼近相同的方式得出参考点。
Path	0 mm	恒速运行区域起点没有位置偏移

型号：

参照点 5 的恒速运行区域的起点有位置偏移。

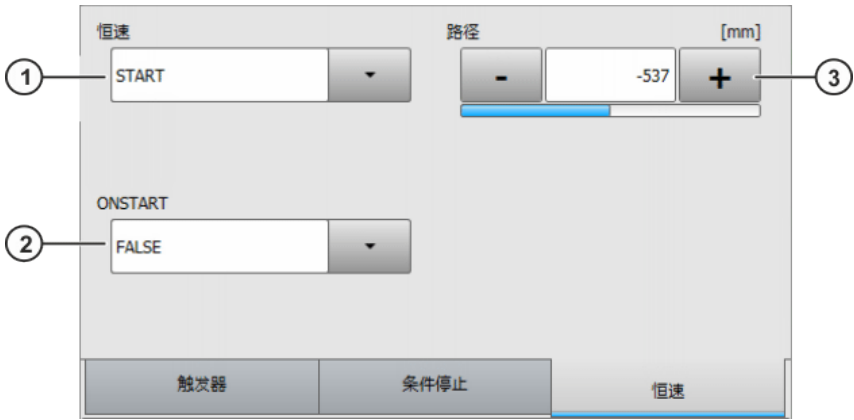


图 9-48：起点已移动

参数	值	说明
恒速	启动	恒速运行区域起点
ONSTART	FALSE	启动指的是目标点 如果目标点已被轨迹逼近，则 Start 或 End 以轨迹逼近弧线的起点为参照。
Path	-537 mm	恒速运行在达到目标点前 537mm 处开始。

- 在样条轨迹上借助程序键向前或向后移动机器人，直到达到了恒速运行的起点。
- 通过按键 **样条动作 > 记录恒速运行区域 Path (轨迹)** 将当前位置应用到 Path 栏中。



图 9-49：样条动作

- 

如果通过手动运行机器人离开该轨迹，则无法借助轨迹进行记录。此处需要重新选择语句。
- 

通过示教将 ONSTART 始终设定为 FALSE 并指定相对于目标点（负）的位置偏移。
- 

原则上，可以用 ONSTART 将相对于起点的偏移设定为 TRUE。但是，在此只能借助输入数字设定偏移量（正）。

6. 在点 7 上结束恒速运行

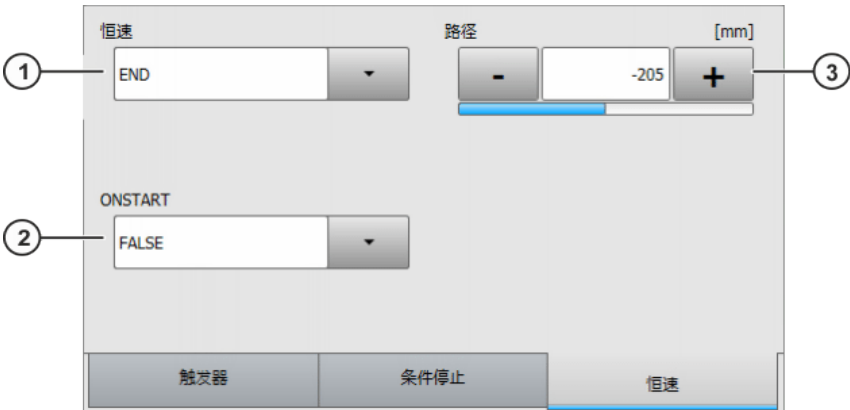


图 9-50：结束恒速

- 该操作步骤相当于记录起点。
- 在此，要在恒速栏（1）中选择 END。
- 在此也可以通过 样条动作 > 记录恒速运行区域 Path（轨迹）移动该点的位置并进行记录。



或者也可以直接在轨迹下输入距离。

检查恒速运行

在运行方式 T1 下测试编程的恒速运行。
尽管机器人以低速离开样条轮廓，它将通过相应的信息报告在恒速运行中与最大编程的速度不同时可以达到哪个速度。



图 9-51：降速

行号表示在此哪个点是限制的大小。



为此，所有折叠必须已打开，并且在视图中还有 DEF 行、详情显示 (ASCII) 已激活，以便可以一一对应行号。



通过有针对性地优化点（位置或姿态）可以最大化待达到的速度。



必须避免直接在 T2 或自动模式下进行初始测试，因为在此会对每个节点输出一条信息。这不适用于优化各个运动点。

9.5.3.2 编程提示

恒速运动区域中的语句选择



如在恒速运动区域内进行语句选择，则机器人控制系统忽略此选择并发出相关信息提示。运动执行情况就与没有编程设定恒速运动区域时一样。



在由偏差值定义的轨迹段中语句选择可视作在恒速运动区域内进行语句选择。然而，在哪个运动组给此区域的起始端和末端编程都无关紧要。

什么可视作语句选择？

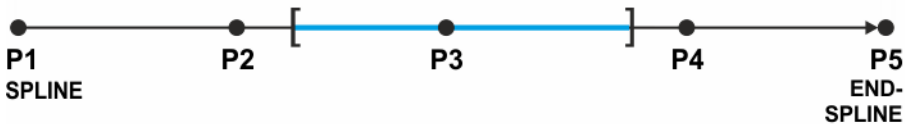


图 9-52：恒速运动区域示例（轨迹）

点上的语句选择...	P1	P2	P3	P4
= 在恒速运动区域中？	否	否	是	否

最大极限值

- 如果样条组的起始点或目标点是精确暂停点：
 - 恒速运动区域最早在起始点处开始。
 - 恒速运动区域最迟在目标点处结束。
- 如果移动值会超出该极限值，则机器人控制系统自动减小偏量，并显示以下信息提示： *CONST_VEL {起始/终止} = {偏量} 无法实现，{新偏量} 即被使用。*
- 机器人控制系统减小偏量，以便形成能够恒定保持编程设定的速度的区域。这就是说：机器人控制系统不一定将极限准确移动至样条组的起始点或目标点，而是可能会往里移。
- 如果该区域从一开始就位于样条组中，但因偏量而无法保持设定的速度，也会显示相同的信息提示。然后，机器人控制系统也会减小偏量。
- 如果样条组的起始点或目标点已被轨迹逼近：
 - 恒速运动区域最早在起始点的轨迹逼近弧线的起点开始。
 - 恒速运动区域最迟在目标点的轨迹逼近弧线的起点结束。

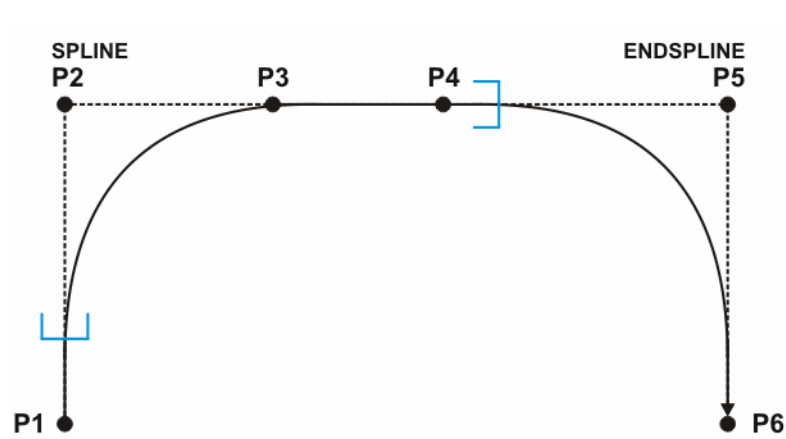


图 9-53：轨迹逼近样条/终端样条时的最大极限值



如果偏量会超出该极限值，则机器人控制系统将极限自动设置为相关轨迹逼近弧线的起点。机器人控制系统不显示信息提示。

9.5.4 练习：给样条运动配设逻辑

练习内容

打开课程练习本并完成以下练习：

- 给样条运动配设逻辑

9.5.5 练习：恒速运动区域和条件停止

练习内容

打开课程练习本并完成以下练习：

- 恒速运动区域和条件停止

10 使用应用程序包

10.1 学习单元：使用工艺程序包

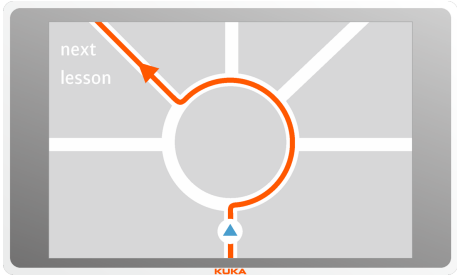


图 10-1：导航器

本学习单元讲授以下内容：

- 了解 KUKA.GripperTech
- 操作 KUKA.GripperTech
- 配置 KUKA.GripperTech
- 编程 KUKA.GripperTech

10.2 了解 GripperTech

说明

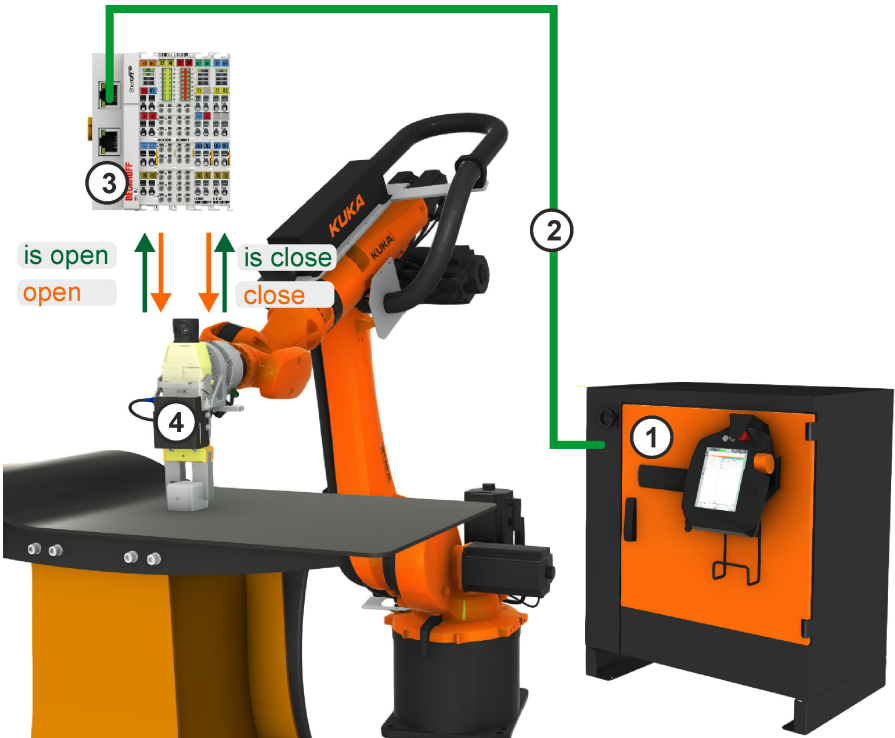


图 10-2：抓手控制

序号	说明	序号	说明
1	KR C4	3	待输入和输出模块的现场总线模块
2	现场总线	4	带数字输入和输出端的抓手

College 培训间示例上的配置

打开抓爪	\$OUT[25]	抓手已打开	\$IN[25]
关闭抓手	\$OUT[26]	抓手已关闭	\$IN[26]

- 在程序运行时，必须打开或闭合工具，例如培训间（3）的抓手。
- 通过已配置的输出端（\$OUT）获得状态变化。
- 工具通过输入端（\$IN）向控制系统反馈当前状态。
- 通常，通过已配置的现场总线系统借助数字输入和输出模块（2）传输给抓手。
- 在此，KUKA.GripperTech 是编程和配置的辅助工具。

下面，处理以下内容

- 通过 smartPAD 操作抓手
(>>> 10.3 “用 KUKA.GripperTech 操作抓手” 页面 296)
- 在 smartPAD 上显示抓手信号。
(>>> 10.3.1 “在 smartPAD 上显示抓手信号” 页面 297)
- 配置抓手
(>>> 10.4 “配置抓手” 页面 299)
- 给抓手进行参数设置
(>>> 10.4.1 “给抓手进行参数设置” 页面 304)
- 借助行指令编程抓手。
(>>> 10.5 “借助行指令编程抓手” 页面 306)

10.3 用 KUKA.GripperTech 操作抓手

说明

在 smartPAD 上有 4 个独立的状态键可进行简单的抓爪或点焊钳操作。客户可以借助配置调整状态键。在 smartPAD 上、状态键右边有一个表明其功能的符号。

操作步骤

显示状态键



图 10-3：显示状态键

- 如果在重新安装或冷机起动后无状态键可用，则可以手动显示状态键。
- **菜单路径：** 机器人按键 > 配置 > 状态键 > Gripper-/SpotTech 或 ServoTech
(视所用技术而定)

操作抓手

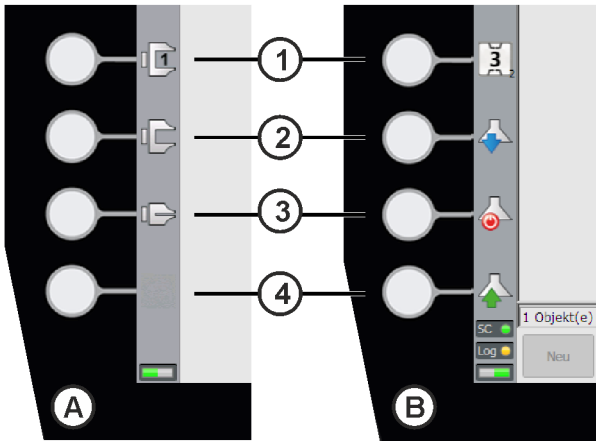


图 10-4：抓爪按键

- 通过状态键（1）可以通过多次按下在预配置的抓手之间切换。
- 图 A：标准抓手**
带打开（2）和闭合（3）功能状态键的抓手。
为此同时按下使能开关。“激活”该图标。
- 图 B：吸持器**
带吹气（2）、关闭（3）和吸气（4）功能状态键的抓手。
为此同时按下使能开关。“激活”该图标。



警告

警告！
使用抓手系统时有挤伤和割伤的危险。抓爪操作人员必须确保身体任何部分都不会被抓爪挤伤。

10.3.1 在 smartPAD 上显示抓手信号

说明

借助抓手显示可以在 smartPAD 上可视化当前的抓手状态。以此可进行安全的操作，尤其是在复杂的抓手（例如吸持器）时。

操作步骤

- 菜单路径：** 机器人按键 > 显示 > Gripper-/SpotTech



图 10-5：抓爪信号的显示

2. 示例：COLLEGE 抓爪

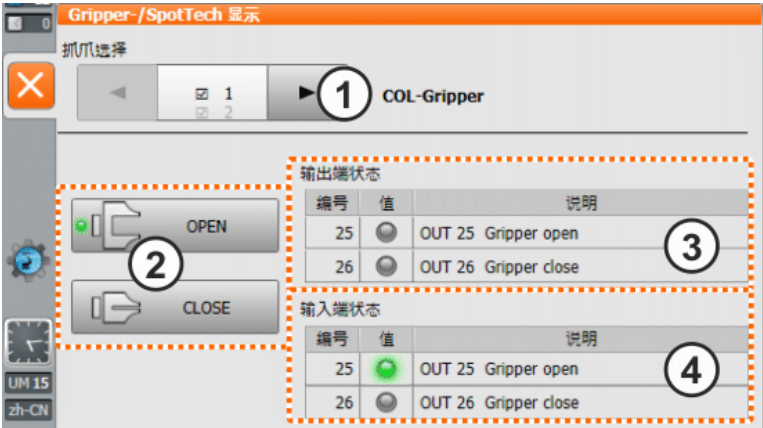


图 10-6：显示抓爪信号

序号	说明
1	选择待显示的抓爪
2	打开和关闭的开关状态 通过按钮可以替代抓爪键操作抓爪。 通过状态键上的 LED 图标显示最后设定的开关状态： - 绿色：开关状态正常 - 红色：开关状态不正常，有故障的输入端同样通过红色 LED 图标显示
3	输出端状态 25 灰色：未按下抓爪键“打开”> 输出端 FALSE 26 灰色：未按下抓爪键“关闭”> 输出端 FALSE
4	输入端状态 25 绿色：抓爪状态为已打开 > 输入端 TRUE 26 灰色：抓爪未关闭（无工件）> 输入端 FALSE

3. 吸持器的示例



图 10-7：吸持器的抓手信号

10.4 配置抓手

说明

KUKA.GripperTech 支持以下功能：

- 控制不同类型的抓手
- 询问和显示当前的抓手位置。
- 确定控制特定抓手位置的条件。（不包含在本次培训中）
- 编程自带用户逻辑（不包含在本次培训中）
- 在控制系统中可对多达 16 种不同的抓手进行配置。

操作步骤

菜单路径：机器人按键 > 配置 > Gripper-/SpotTech > 抓手

主菜单	配置	Gripper-/SpotTech
文件	输入/输出端	抓爪
配置	SUBMIT 解释器	参数
显示	状态键	
诊断	用户组	
投入运行	其它	
关机	安全配置	
帮助	机器配置	
Runtime Tests	Gripper-/SpotTech	

图 10-8：菜单路径

选项卡：抓爪

- 在选项卡抓爪 (A) 中配置抓爪特定的设置。
- 在随后出现的选项卡状态和输入输出端中定义在此规定的开关状态以及输入和输出端。



图 10-9：选项卡：抓爪

参数	说明
抓爪选择 (1)	确定待配置的抓爪。 有 16 个保存位置可用
抓爪名称 (2)	抓爪的名称（最多 24 个字符） 名称显示在指令行中。预定义的名称可更改。
状态 (2)	- 勾选：可以通过状态键操作抓爪和通过 ILF 对抓爪进行编程设定。 - 未勾选：抓爪未激活。
开关状态的数量 (2)	抓爪开关状态的数量 1 ... 6
输出端数量 (2)	用于控制抓爪的输出端的数量。 1 ... 6
输入端数量 (2)	用于监控抓爪的输入端的数量。 1 ... 6
抓爪图标 (2)	选择状态键和显示插件的图标 默认：带数字的图标

参数	说明
开关状态超时 (2)	在状态检查时，最长等待直至该开关状态必须存在的时间（带监控的 Set Gripper, Check Gripper）。 如果该时间用完并且该开关状态仍不存在，则启动故障策略。 如果该开关状态存在，则继续程序，而无需等待该时间。 0…10 s 默认：3 s
输出端的脉冲持续时间 (2)	作为脉冲定义的输出端的脉冲持续时间。 0…3 s 默认：0.5 s



借助参数旁的箭头符号可显示详细说明。
作为示例，请见插图开关状态超时 [s] 和输出端脉冲时长 [s]。

选项卡：状态和输入输出端

- 在选项卡**状态和输入输出端 (B)** 中分配开关状态 (2) 以及输出端 (3) 和输入端 (4)。
- 在此分配为输入和输出端定义的开关状态以及手动操作的前提条件。

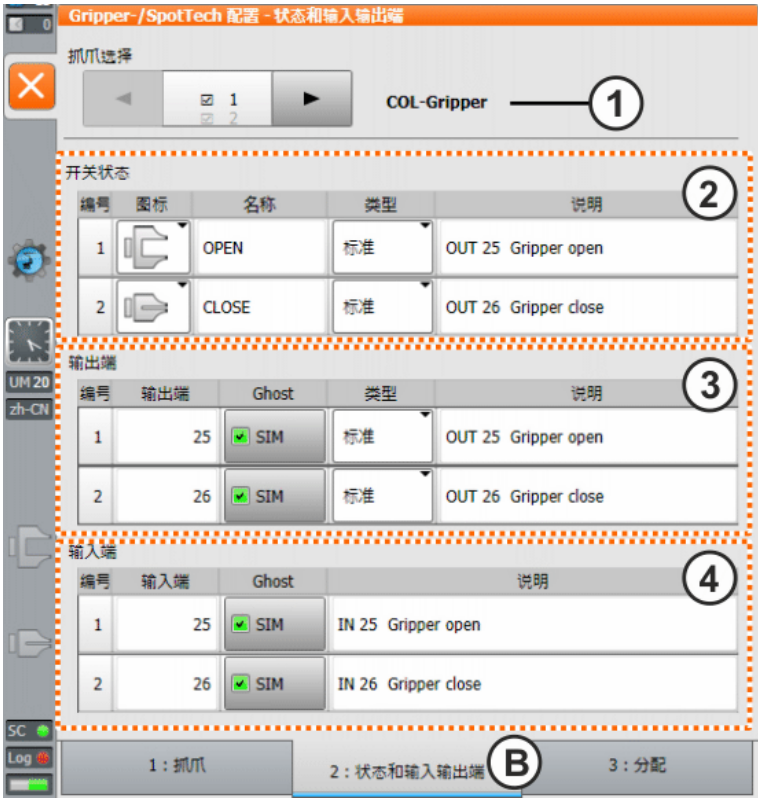


图 10-10：选项卡：状态和输入输出端

- 开关状态 (2)

参数	说明
编号	开关状态的编号 控制系统的内部索引以 1 开头且连续。 无法更改
图	开关状态图标 选择要在状态键上和开关状态显示插件中显示的图标。
名称	开关状态的名称 如果选择了相关的抓手，则在行指令中显示。
类型	用于该开关状态的状态键的类型 - 标准键：立即执行动作。 - 临界按键：为了执行动作，必须在 4 s 内连续按下按键 2 次。只有前 3 个状态键可定义为临界按键。剩余的 3 个状态键只能在显示插件中显示，因此不可定义为临界键。
说明	开关状态的说明

• 输出端 (3) 和输入端 (4)

参数	说明
编号	输出端控制的内部索引 输入端分析的内部索引 以 1 开始并且连续编号。 无法更改
输出端/输入端	输出端和输入端 默认：0
Ghost/SIM	是否要在 Ghost 模式下设定或分析输入/输出端的设置。 输入端： - Sim, 勾选：不分析输入端。 - Sim, 未勾选：在 Ghost 模式下也分析输入端。 输出端： - Sim, 勾选：不设定输出端。 - Sim, 未勾选：在 Ghost 模式下也设定输出端
类型	输出端的类型 - 标准：将该输出端定义为标准输出端。状态保持不变。 - 脉冲：将该输出端定义为脉冲输出端。设定的状态只为规定的时间保持不变。
说明	输出端和输出端的长文本

选项卡：分配

- 在选项卡**分配 (3)** 中给定义的开关状态分配定义的输入和输出端以及手动操作的前提条件。
- 为了在该选项卡中有开关状态以及输入和输出端可用，必须在前面的选项卡**状态和输入输出端**中对其进行定义。

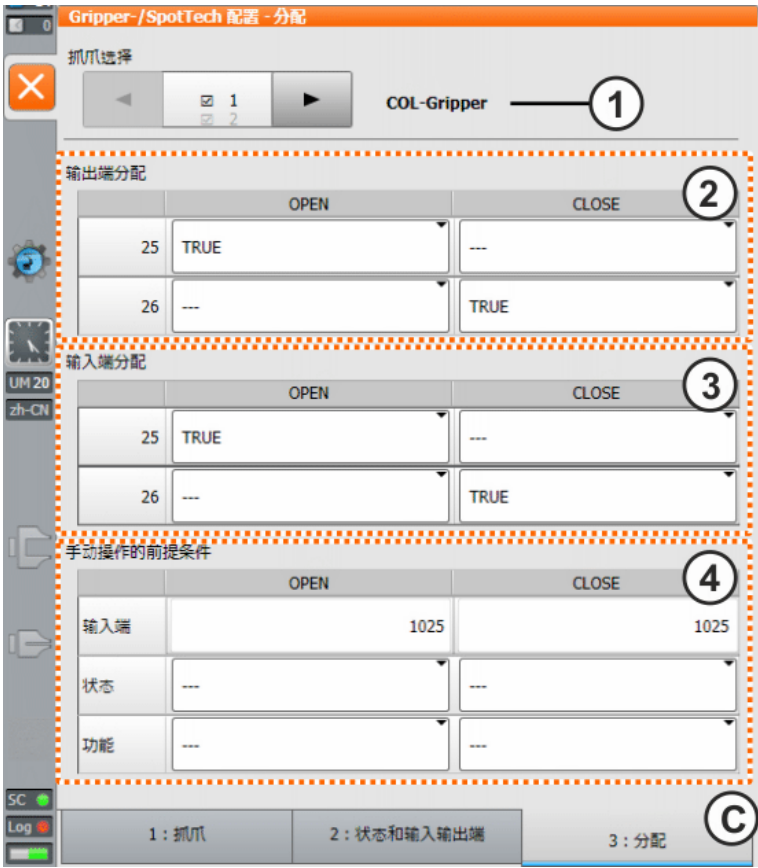


图 10-11：选项卡：分配

分配	说明
输出端分配 (2)	为一个状态设定输出端 - TRUE: 将输出端设定为 TRUE。 - FALSE: 将输出端设定为 FALSE。 [空]: 不设定输出端，当前状态保持不变。
输入端分配 (3)	为一个开关状态分析输入端 - TRUE: 输入端必须已设定。 - FALSE: 输入端不允许已设定。 [空]: 输入端不相关并且不进行分析。

分配	说明
手动操作的前提条件 (4)	可以为手动设定开关状态最多规定 3 个前提条件。只在满足了规定的前提条件时才设定该开关状态。 输入端: - 规定输入端的编号。 - 定义的输入端必须为 TRUE。 - 默认: 1025 状态: - 选择定义的开关状态 - 如果在“状态和输入/输出”选项卡中存在定义的开关状态, 则设定新的开关状态。 - 默认: [空] 功能: - 选择最多 10 个不同的用户程序。可在 Grp_User.src (R1\TP\GripperSpotTech) 中对它们进行编程设定。每个条件发回一个返回值。 - 返回值: TRUE: 设定开关状态。 FALSE: 不设定开关状态。 - 默认: [空]

10.4.1 给抓手进行参数设置

说明

Ghost 操作

如果将一个设备投入运行, 或在一个生产环节结束之后将设备“清空运行”, 则通常在无部件时进行。在配置中清空运行被称为 **Ghost 运行**。为了预防意外的抓爪动作, 例如抓爪拾取一个部件, 可以在配置中明确地将其停用。对于吸持器, 通常使用其他的方法。在此, 用吹气状态替代吸气状态控制抓爪。调换状态。在拾取金属板时, 以此防止滑动或移动。可以手动通过配置菜单激活和停用 Ghost 运行。或者也可以通过 PLC 的输入和输出端接通和关闭 Ghost 运行。

操作步骤

抓手参数

- 菜单路径: 机器人按键 > 配置 > Gripper-/SpotTech > 参数



图 10-12: 菜单路径

- 抓手参数



图 10-13: 抓手参数

参数	说明
Ghost 模式接通输入端 (1)	接通 Ghost 模式的输入端编号 — 0: 未通过输入端开关 Ghost 模式。 — 1 ...4096/8192: 如果输入端为 TRUE, 则 Ghost 模式已激活。 默认: 0
Ghost 模式接通 (2)	接通 Ghost 模式 (未通过输入端) — 勾选: Ghost 模式已接通。 — 未勾选: Ghost 模式已关闭。 默认: Ghost 模式已关闭。
出错时的输出端编号 (3)	出错时设定的输出端编号 (上级控制系统上的错误信息)。 — 0: 不设定输出端。 — 1 ... 4096/8192: 用该编号设定输出端。 默认: 0

参数	说明
故障策略 1...3 (4-6)	可以配置 3 个故障策略，之后可以在带抓手监控的行指令中选择。 如果没有切换状态，则执行所选的故障策略。

10.5 借助行指令编程抓手

说明

用工艺程序包 KUKA.GripperTech 可通过预制的指令行直接在所选的程序中对抓手指令进行编程。在此有两条控制抓爪的指令可供使用。

- **Gripper SET**
在程序中用于打开和闭合 College 抓爪的指令。
- **Gripper SYN SET**
在程序中用于打开和闭合 College 抓爪的指令。
此外，可以在时间上以及基于轨迹推移开关点。

抓手编程功能

原则上可为抓手指令相应编程，使指令相对起始点或目标点得到执行。

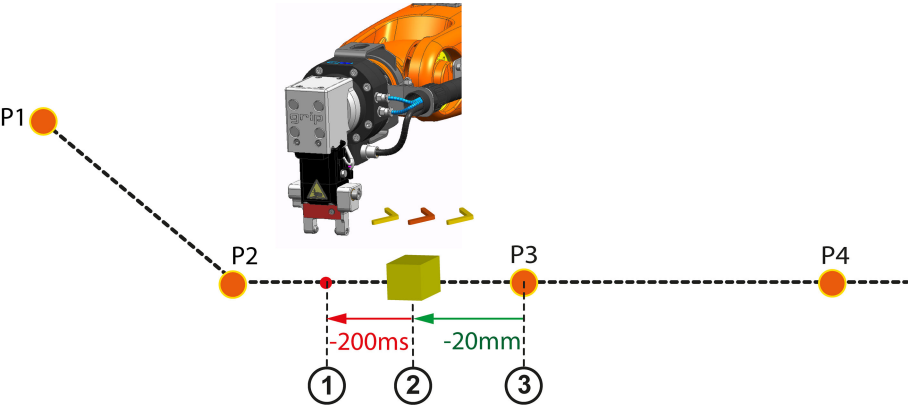


图 10-14：带位置移动和时间推移的抓爪函数示例

序号	说明
1	打开抓爪
2	抓手位置
3	编程设定的点

- 示例中，要抓取的方块（2）位于编程设定的点 P3（3）前面 20mm 处。
- 在达到方块之前 200ms（1），机器人抓手应该打开。



警告

选择运动中具有加工的抓手指令时一定要格外小心，因为考虑不周时飞出的工件或发生碰撞会造成人员或财产的损伤！

操作步骤

1. 在所选的程序中选中要在下方插入抓爪指令的程序行。

```
1 INI
2
3 SPTP HOME Vel=100% % DEFAULT
```

```
4 SLIN P1 Vel=2m/s CPDAT1 Tool[1] Base[1]
5 SLIN P2 Vel=0.2m/s CPDAT2 Tool[1] Base[1]6
6 ;#### 在此插入抓手命令 ###
7 SLIN P3 Vel=0.2m/s CPDAT3 Tool[1] Base[1]
8 SLIN P4 Vel=2m/s CPDAT3 Tool[1] Base[1]
9 ...
```

2. 菜单路径：命令 > GripperTech > Gripper....
插入所需的抓手命令。

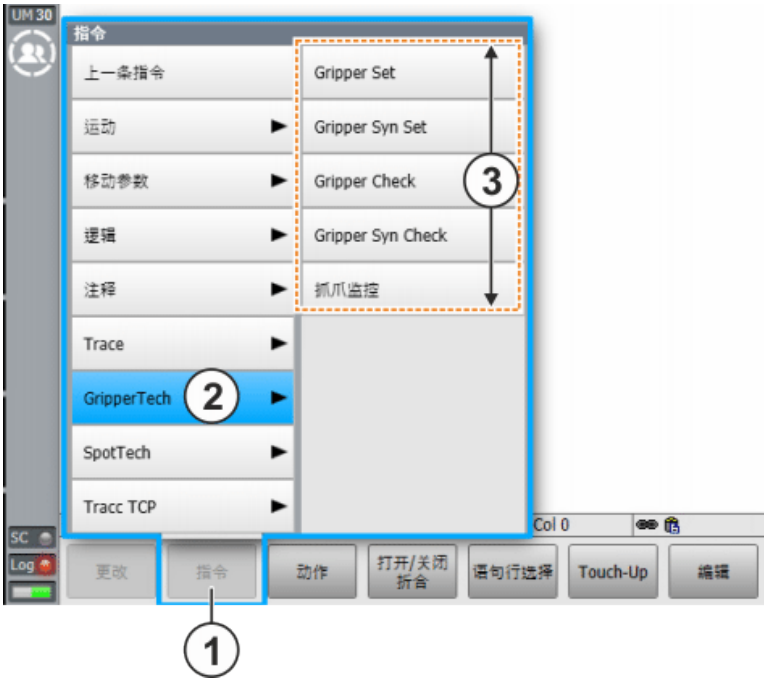


图 10-15：抓手命令

3. 在行指令中设置所需的抓爪状态、等待时间或错误策略。
示例：Gripper SET

```
1 INI
2
3 SPTP HOME Vel=100% % DEFAULT
4 SLIN P1 Vel=2m/s CPDAT1 Tool[1] Base[1]
5 SLIN P2 Vel=0.2m/s CPDAT2 Tool[1] Base[1]
6 ; ### 抓手命令 ###
7 Gripper SET [1]Gripper State=[1]OPEN Wait 0.2[s]Check with No Check
7 SLIN P3 Vel=0.2m/s CPDAT3 Tool[1] Base[1]
8 SLIN P4 Vel=2m/s CPDAT3 Tool[1] Base[1]
9 ...
```

Gripper SET

- 通过该行指令设定抓爪的开关状态。
- 可以在主进和预进中切换开关状态。
- 在主进中使用时，可以在抓爪反应后等待一段确定的时间，直到到达下一个运动语句。

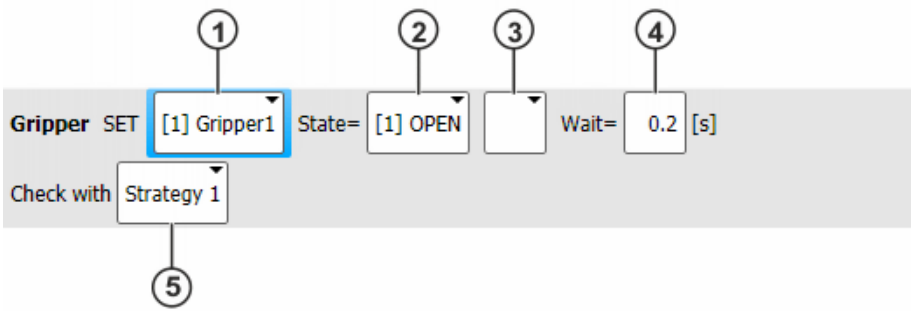


图 10-16: Gripper SET 行指令

序号	说明
1	已激活的抓爪 只显示激活的抓爪。
2	所设定的抓爪的开关状态。 可选择开关状态的数量和其名称取决于配置。
3	• CONT: 通过预进设定开关状态。 • [空]: 用主进设定开关状态。
4	等待时间 如果指定的等待时间结束，则继续程序。 • 0.0 ... 10.0 s 默认: 0.2 s 只在用主进切换时，才显示该栏。
5	故障策略 如果一个状态不存在，则可为其设置一个故障策略。 • No Check: 继续编程，而不检查是否有该开关状态。 • Strategy 1...3: 配置的故障策略

Gripper SYN SET

- 用该行指令可以在运动的起点或目标点上设定抓手的开关状态。
- 开关状态在时间和/或位置上均可推移。

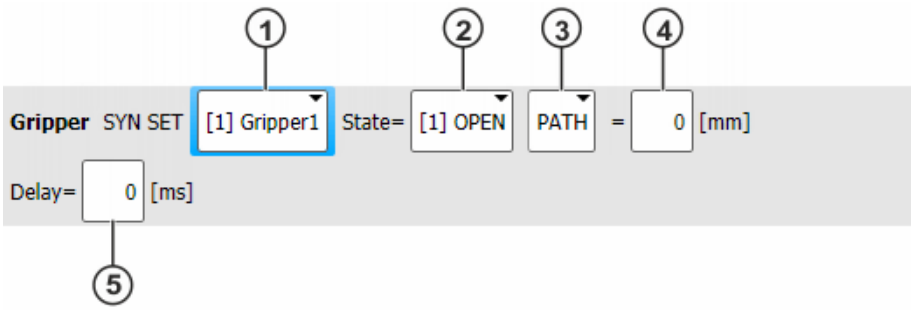


图 10-17: Gripper SYN SET 行指令

序号	说明
1	已激活的抓手 只显示激活的抓手。
2	所设定的抓手的开关状态 可选择开关状态的数量及其名称取决于配置。
3	以 Gripper SYN SET 为参照的点 • START: 时间推移的起始参考点 • END: 时间推移的目标参考点 • PATH: 开关状态的设定以运动的目标点为参照。除了位置移动外，还可以进行时间推移。
4	如果选择了 PATH ，则只显示该栏。 切换点至目标点的距离 • -2,000 ... +2,000 mm
5	输出动作的时间偏移量 • -1000 ... 1000 ms 此时间数值为绝对值。视机器人速度不同，输出点将随之变化。

SYN SET 示例

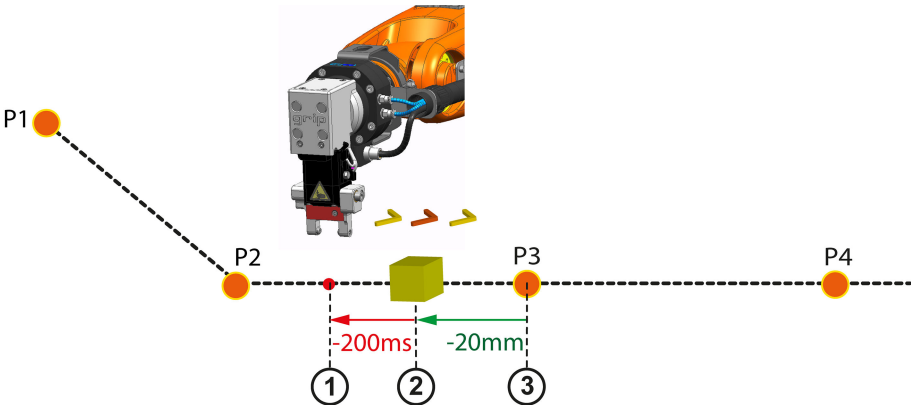


图 10-18：带位置移动和时间推移的抓爪函数示例

```
1 INI
2
3 SPTP HOME Vel=100% % DEFAULT
4 SLIN P1 Vel=2m/s CPDAT1 Tool[1] Base[1]
5 SLIN P2 Vel=0.2m/s CPDAT2 Tool[1] Base[1]
6 ; *** 打开抓爪 ***
7 Gripper Syn SET [1]Gripper State=[1]OPEN Path=-20mm Delay=-200ms
8 ; *** 闭合抓爪 ***
9 Gripper Syn SET [1]Gripper State=[1]CLOSE Path=-20mm Delay=0ms
10 SLIN P3 Vel=0.2m/s CPDAT3 Tool[1] Base[1]
16 ...
```

10.5.1 借助行指令检查抓爪状态

说明

用工艺程序包 KUKA.GripperTech 可通过预制的指令行直接在所选的程序中检查抓爪状态。

有两条指令可供使用

- **Gripper CHECK**
检查至主运行或预进中的开关时间。
- **Gripper SYN CHECK**
开关时间的检查可以在时间上以及位置上推移。

操作步骤

Gripper CHECK

通过该行指令检查抓爪的开关状态。

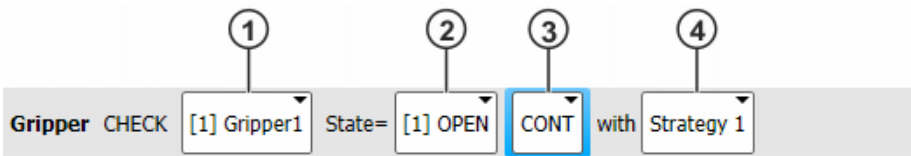


图 10-19: Gripper CHECK 行指令

序号	说明
1	已激活的抓爪 只显示激活的抓爪。
2	所检查的抓爪的开关状态 可选择开关状态的数量和其名称取决于配置。
3	• CONT: 通过预进检查开关状态。 • [空]: 用主进检查开关状态。
4	故障策略 如果一个状态不存在，则可为其设置一个故障策略。 • No Check: 继续编程，而不检查是否有该开关状态。 • Strategy 1...3: 配置的故障策略

Gripper SYN CHECK

- 用该行指令可以在运动的起点或目标点上检查抓手的开关状态。
- 开关状态的检查可以在时间和/或位置上推移。

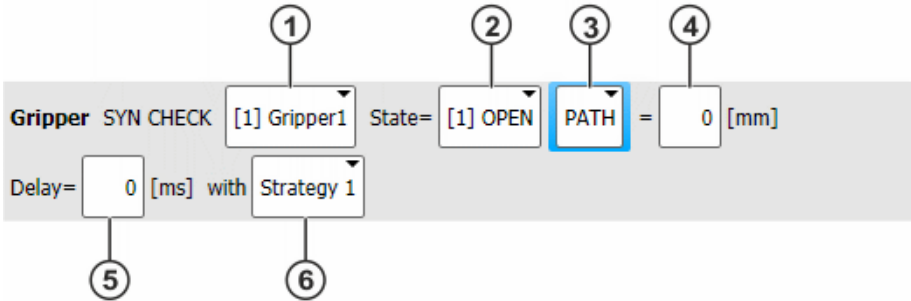


图 10-20: Gripper SYN PULSE 的行指令

序号	说明
1	已激活的抓手 只显示激活的抓手。
2	所检查的抓手的开关状态 可选择开关状态的数量及其名称取决于配置。
3	以 Gripper SYN CHECK 为参照的点 • START: 以起点为参照进行时间推移 • END: 以目标点为参照进行时间推移 • PATH: 检查以运动的目标点为参照。除了时间推移外，还可以进行位置移动。
4	如果选择了 PATH ，则只显示该栏。 切换点至目标点的距离 • -2,000 ... +2,000 mm
5	检查的时间偏移量 • -1000 ... 1000 ms 此时间数值为绝对值。即视机器人速度不同，检查的时间将随之变化。
6	故障策略 如果一个状态不存在，则可为其设置一个故障策略。 • No Check: 继续编程，而不检查是否有该开关状态。 • Strategy 1...3: 配置的故障策略

10.5.2 练习：抓爪编程“标牌”

练习内容

打开课程练习本并完成以下练习：

- 抓爪编程“标牌”

10.5.3 练习：抓爪编程“笔”

练习内容

打开课程练习本并完成以下练习：

- 抓爪编程“笔”

11 外部工具的配置和编程

11.1 学习单元：外部工具的配置和编程

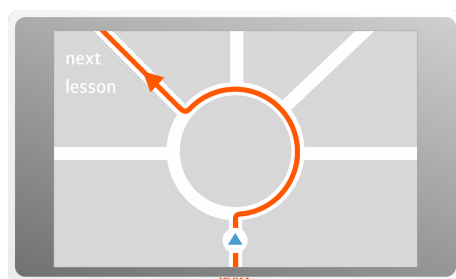


图 11-1: 导航器

本学习单元讲授以下内容：

- 机器人的调试
 - 固定工具的测量
 - 测量由机器人引导的工件
- 固定工具点动
- 用外部 TCP 进行运动编程

11.2 机器人引导的工件和固定工具

说明

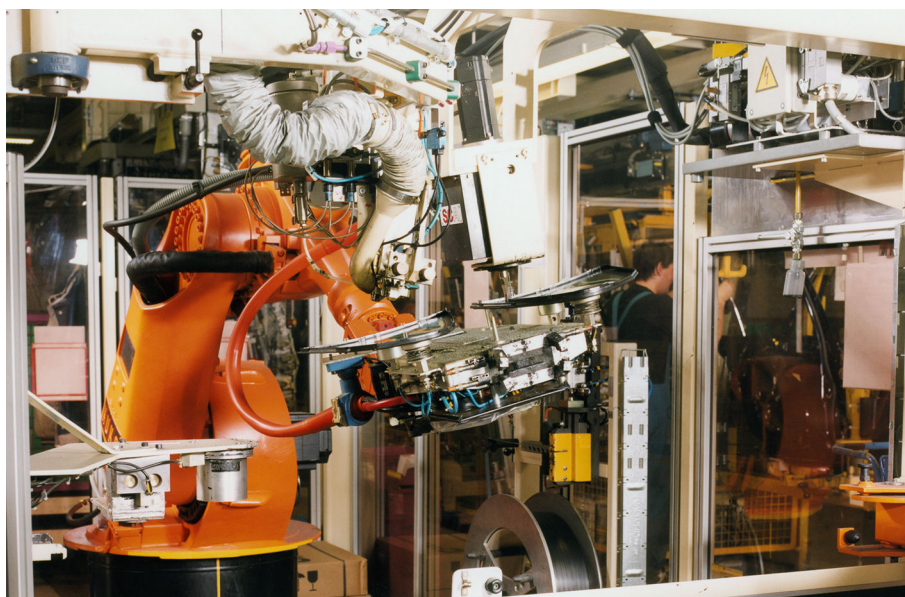


图 11-2: 粘贴侧面玻璃



在该示例中，侧面玻璃由机器人引导，并且粘胶被涂抹到固定工具上。

- 由于工艺特定的原因，可能需要：
 - 待加工的工件必须由机器人引导，例如便于实现优化的运动过程。
 - 工具例如由于结构尺寸而必须固定放置。
- 通过使用固定工具和机器人引导的工件：
 - 工具重量不再重要。

- 介质供应更容易实现。
也可以使用不适合拖链用途的供电线路。
- 需要下列测量步骤：
 - 测量固定工具
(>>> 11.2.1 “测量固定工具” 页面 316)
 - 测量机器人引导的工件
(>>> 11.2.2 “测量由机器人引导的工件” 页面 321)

11.2.1 测量固定工具

说明

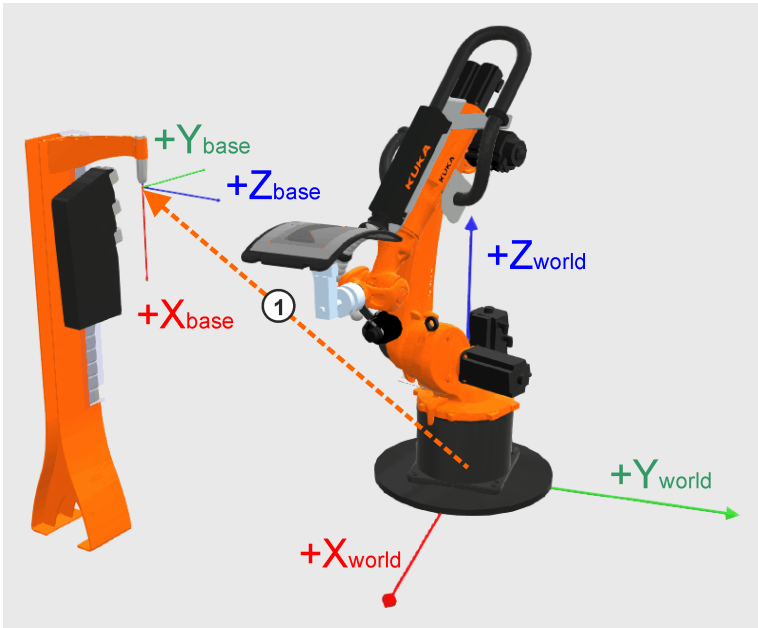


图 11-3：测量固定工具基础

1 外部 TCP 基于世界坐标系的距离 (XYZ) 和朝向 (ABC)



基于世界坐标系 (\$WORLD) 的外部 TCP (1) 用作基坐标系。

固定（外部）工具的测定分为 2 步：

1. 确定外部 TCP 相对于世界坐标系原点的位置
2. 根据外部 TCP 确定该坐标系方向

前提条件

- 在测量之前，法兰坐标系的方向必须已知。
- 测量外部 TCP 时需要一个由机器人引导的已测工具。
- 这尤其适用于使用 6D 法时，因为基坐标（外部 TCP）的 +Z 从 +X 法兰得出。



在测量之前，调整工具 0（法兰）和工具坐标系。沿 +X 方向运行并在法兰上标记方向。

操作步骤

确定外部 TCP

1. 通过工具和基坐标管理添加了一个新的固定工具。

(>>> 4.8.1 “添加新的基坐标/固定工具” 页面 142)

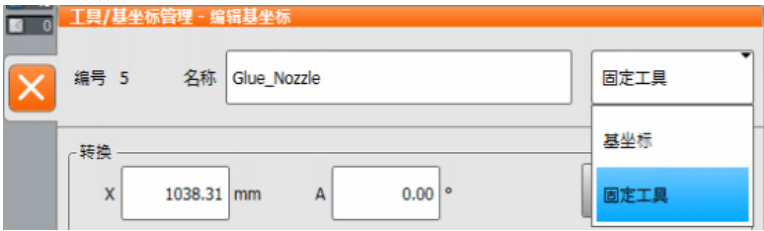


图 11-4：创建固定工具

2. 通过按键**测量**选择 XYZ 方法。



图 11-5：测量方法

3. 将参考工具驶向外部 TCP
用所测量机器人引导型工具的工具顶尖移至外部 TCP。



图 11-6：第一个点：原点，坐标系，外部 TCP

4. 示教外部 TCP



图 11-7：测量 TCP

- 在下拉窗口 (1) 中选择参考工具。
- 如果参考工具的 TCP 和固定工具的 TCP 一致，则点击 TCP 校准点行 (2)。
该行的背景色被置为蓝色。
- 用 Touch-Up 按键 (3) 记录该点。
在转换栏中显示确定的值。
- 在测量结果窗口 (4) 中显示确定的工具偏移量。
- 借助同名按键保存 (5) 保存值。



借助驶向 (PTP) 按键可以反复驶向已示教过的点。

5. 新固定工具的位置

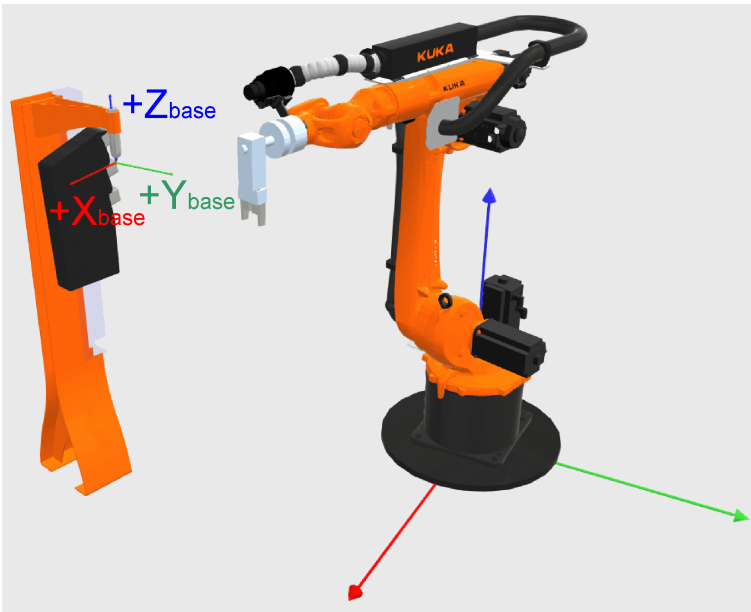


图 11-8：测量的外部工具

确定方向

1. 通过工具和基坐标管理添加了一个新的固定工具。
(>>> 4.8.1 “添加新的基坐标/固定工具” 页面 142)
确定并保存了固定工具的 TCP。
(>>> “确定外部 TCP” 页面 316)
2. 在工具管理中打开该固定工具。
通过**测量**按键选择 **ABC-Welt** 方法。



图 11-9：测量方法

3. 规定作业方向

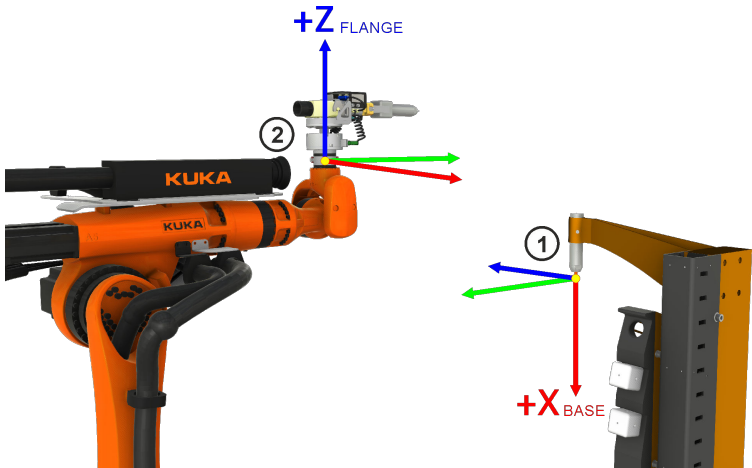


图 11-10：测量固定工具 ABC 世界法 5D

- 1 +X 固定工具
 - 2 +Z 法兰
 - 确定姿态时要将 FLANGE 坐标系校准至平行于新的坐标系。
 - 在该测量时要注意机器人手腕。对于许多机器人类型，可以将机器人手腕用作参考点。
4. 示教作业方向



图 11-11：工具作业方向

- 在 5D/6D 选择窗口 (1) 中确定测量方法
有两种不同方式：
 - 5D：只将固定工具的作业方向告知机器人控制器。该作业方向被默认为 X 轴。其它轴的姿态将由系统确定，对用户来说，不是很容易识别。
 - 6D：所有 3 个轴的方向都告知机器人控制器。
- 点击校准点，ABC 姿态行 (2)。
该行的背景色被置为蓝色。
- 借助按键 Touch-Up (3) 应用机器人位置。
在转换栏中显示确定的值。
- 在测量结果窗口 (4) 中显示确定的工具偏移量。
- 借助同名按键保存 (5) 保存值。



借助驶向 (PTP) 按键可以反复驶向已示教过的点。

11.2.2 测量由机器人引导的工件

说明

测量机器人引导的工件

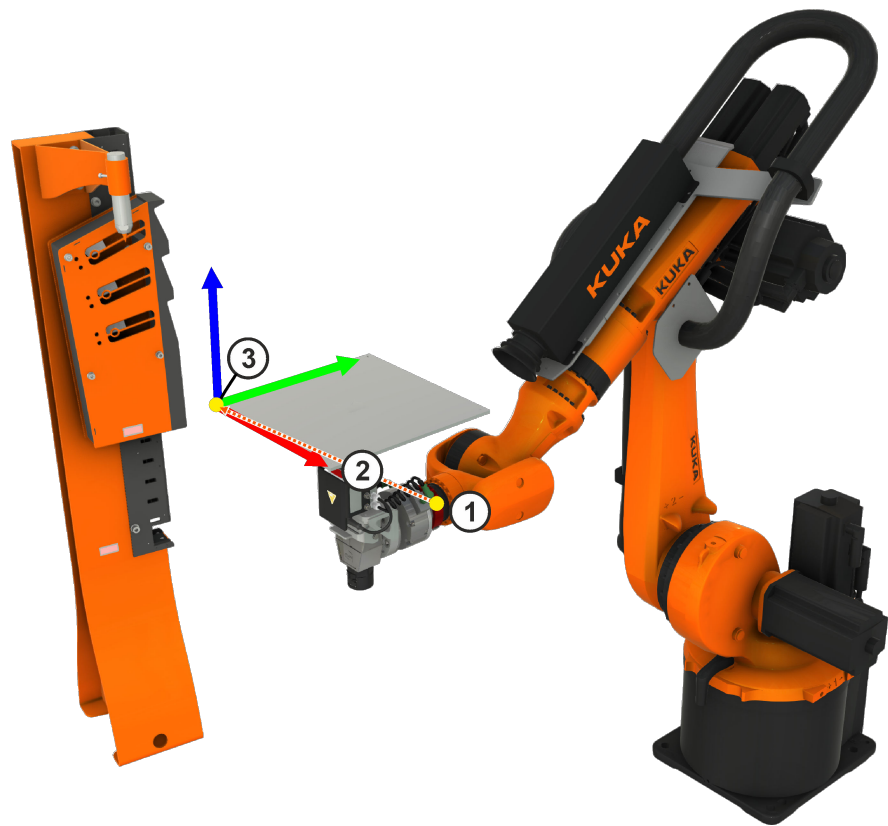


图 11-12：通过直接测量的方法测量工件

- 1 法兰中心点
- 2 1 和 3 和扭转之间的距离
- 3 已测量工件的坐标系原点

专家文件

下文仅讨论直接测量法。间接测量法极少使用，在文件 *KUKA System Software 8.6* 中有说明。



Xpert 搜索: **KUKA System Software 8.6**
面向系统集成商的操作及编程指南
Xpert 筛选器: 文献 > 操作与编程指南
章节: 测量 > 测量方法概览 > 间接法

操作步骤

- 1. 固定工具已测量。

(>>> 11.2.1 “测量固定工具” 页面 316)

2. 借助抓爪等拾取测量所需的工件。
3. 通过工具和基坐标管理添加了一个新的机器人引导工件。

(>>> 4.6.1 “添加新的工具/工件” 页面 118)

4. 在编辑工具窗口的下拉窗口中选择工件。

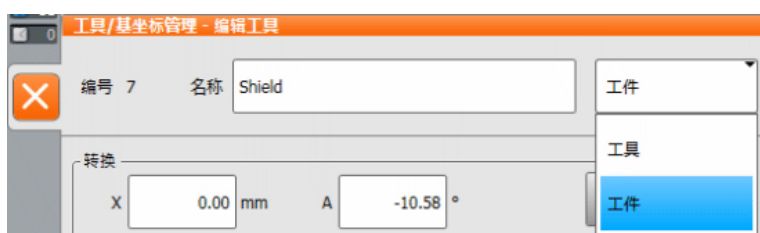


图 11-13: 定义机器人引导的工件

5. 通过按键**测量**选择 3 点方法。



图 11-14: 测量基坐标, 选择

6. 驶向坐标原点

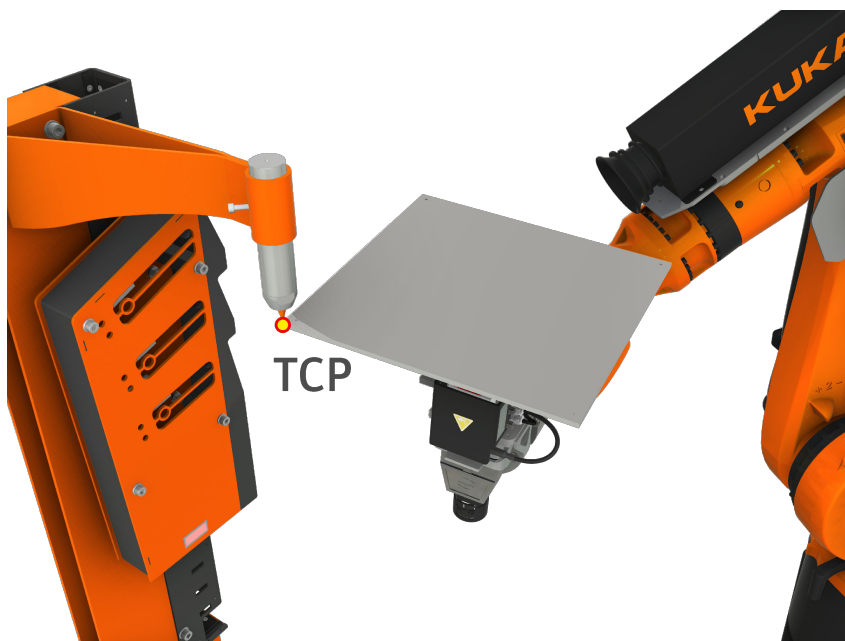


图 11-15: 坐标原点, 工件

- 将工件驶向固定工具的 TCP。
- 在此, 必须使待定义坐标系的原点与固定工具的 TCP 一致。

7. 测量坐标原点

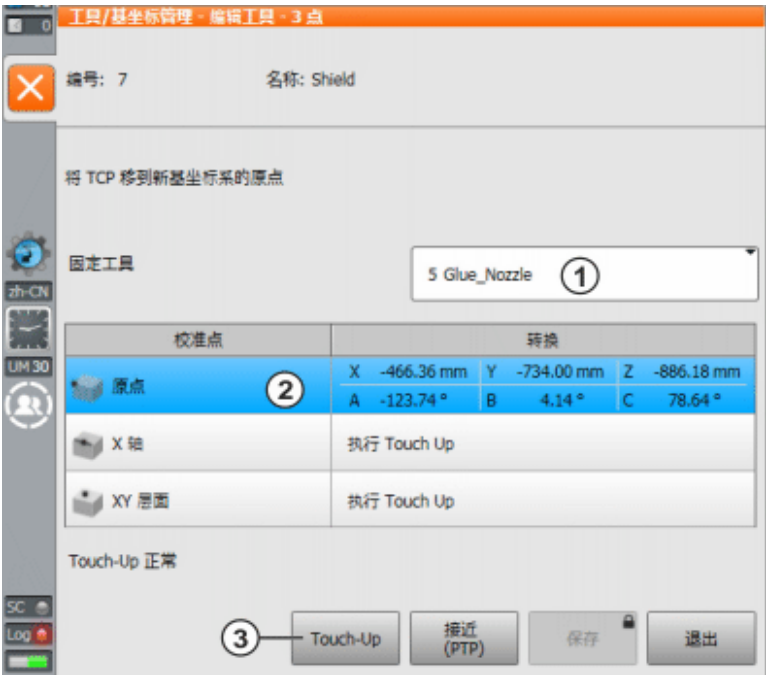


图 11-16：测量坐标原点

- 在下拉窗口 (1) 中选择固定工具。
- 如果坐标原点和固定工具的 TCP 一致，则点击原点校准点行 (2)。
- 用 Touch-Up 按键 (3) 记录该点。
在转换栏 (2) 中显示确定的值。

8. 驶向 X 姿态上的点

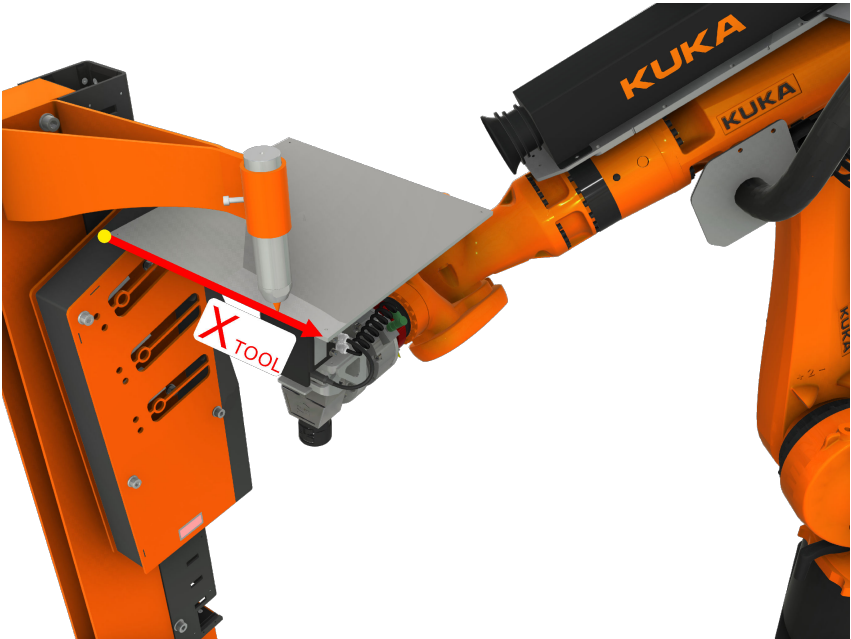


图 11-17：X 姿态

- 将固定工具的 TCP 驶向待定义工件坐标系正 X 轴上的一个点。

9. 示教 X 姿态上的点



图 11-18: 测量 X 轴

- 点击 X 轴校准点 (2) 行。
该行的背景色被置为蓝色。
- 用 Touch-Up 按键 (2) 记录该点。
在转换栏 (2) 中显示确定的值。

10. 驶向 XY 平面上的点

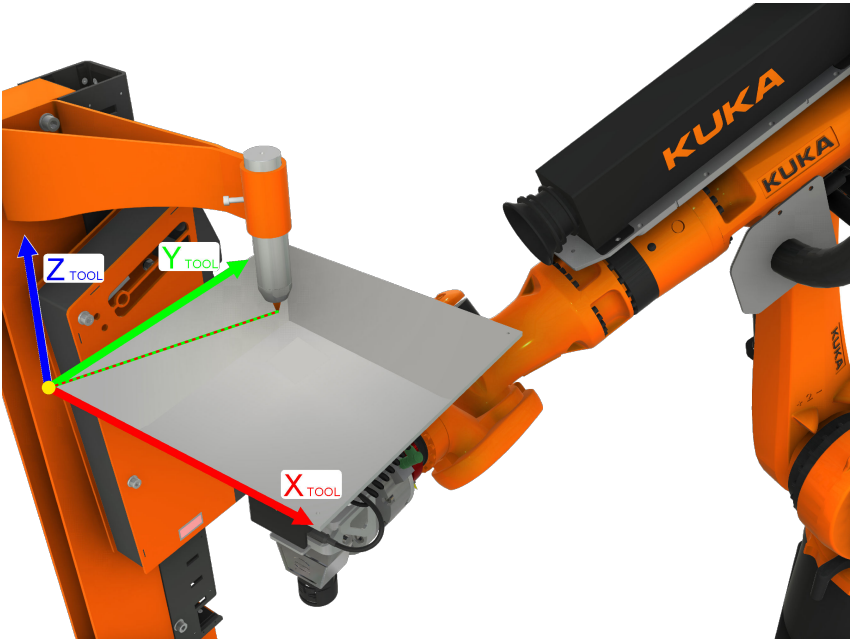


图 11-19: XY 平面

- 将固定工具的 TCP 驶向待定义工件坐标系 XY 平面上的一个点。

11. 示教 XY 平面上的点



图 11-20: 测量 XY 平面

- 点击 XY 平面行 (2)。
- 借助按钮 Touch-Up (2) 记录位置。
在转换栏中显示确定的值。
- 记录最后一个点，表明基坐标的测量完成。在测量结果栏 (3) 中显示偏移量值。
- 用同名按钮保存 (4) 保存参考点的偏移量值和扭转。

11.2.3 练习：测量外部工具和机器人引导的工件

练习内容

打开课程练习本并完成以下练习：

- 测量外部工具和机器人引导的工件

11.3 固定工具点动

说明

优点和应用领域

有些生产和加工过程要求机器人搬运**工件**（部件）(1) 而不搬运**工具**（2）。在我们的示例中，需将粘接剂涂抹在潜水镜的玻璃镜片上。优点是，部件无需先放置好便能加工，因此可节省夹紧工装时间。例如，这适用于以下情况：

- 粘接应用
- 焊接应用

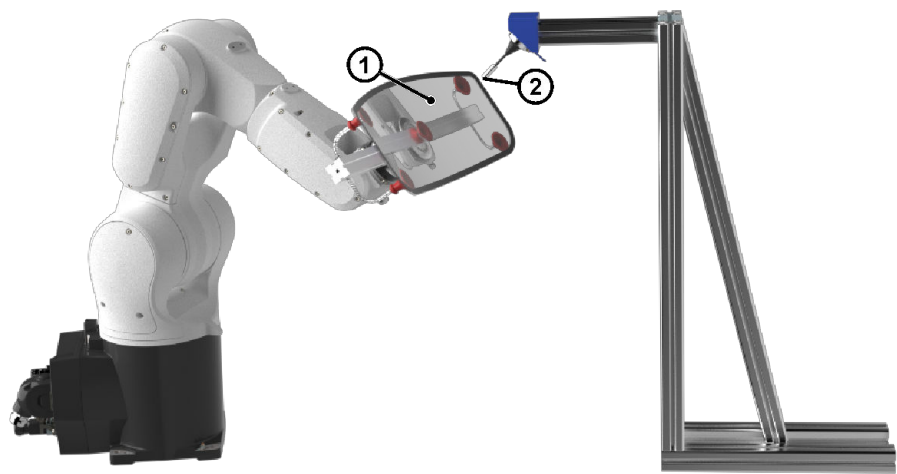


图 11-21：固定工具示例

提示

为对此类应用成功编程，既要测量固定工具的外部工具中心点，也要测量工件。

用固定工具时更改的运动过程

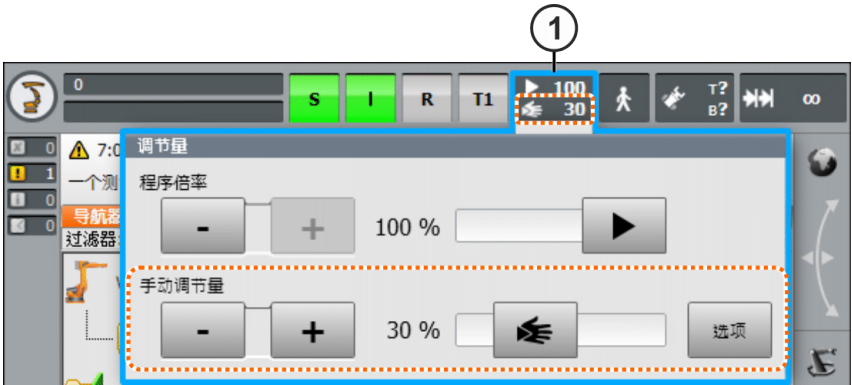
机器人引导型工具

多数情况下，由机器人引导工具并加工或输送工件。针对每个编程的运动语句（示教）将保存使用中的基坐标系与使用中的工具坐标系（TCP）的距离。其中，存储的运动（插补）始终以机器人引导的工具的 TCP 为基准。



图 11-24：在选项菜单中选择外部 TCP

- 在工具选择窗口（2）中激活由机器人引导的工件。
 - 在基坐标选择窗口（3）中激活固定工具。
 - 将插补模式选择设置为**外部工具**（4）。
2. 设定手动倍率（1）。



3. 根据自己的喜好在 smartPAD 上设置手动移动。



图 11-25: 手动移动, 外部工具, 6D 鼠标

- 设定工具, 以便在工件坐标系中移动,
 - 设定基坐标, 以便在外部工具坐标系中移动,
4. 将四个确认键中的一个按至中间位置并按住。



5. 用点动键/6D 鼠标朝所需方向移动。



图 11-26: 手动移动

控制系统通过在**手动移动选项**选项窗口中选择**外部工具**切换：所有运动现在均相对外部 TCP，而不是由机器人导引的工具。

11.3.1 练习：用固定工具手动移动

练习内容

打开课程练习本并完成以下练习：

- 用固定工具手动移动

11.4 用外部 TCP 进行运动编程

用外部 TCP 进行运动编程

用固定工具进行运动编程时，运动过程与标准运动相比会产生以下区别：

- 行指令中的标识：在选项窗口 **Frames** 中，**外部 TCP** 项的值必须为“TRUE”。

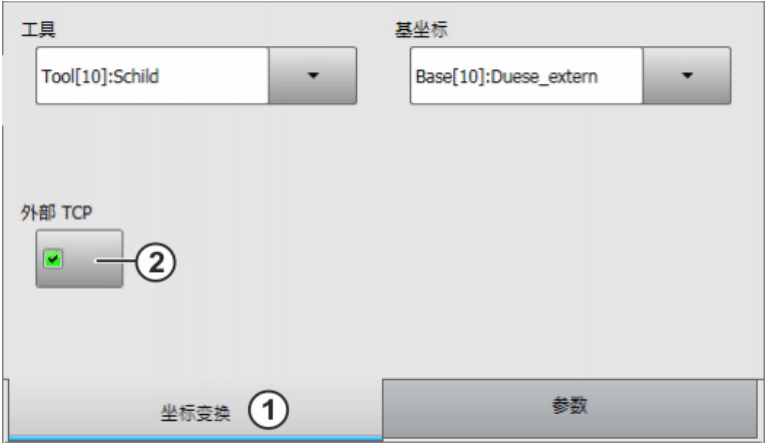


图 11-27：选项窗口“坐标系”：外部 TCP

- 运动速度**以外部 TCP 为基准。
- 沿轨迹的**姿态**同样以外部工具为基准。
- 不但要指定合适的基坐标系（固定工具/外部 TCP），而且也要指定合适的工具坐标系（机器人引导的工件）。

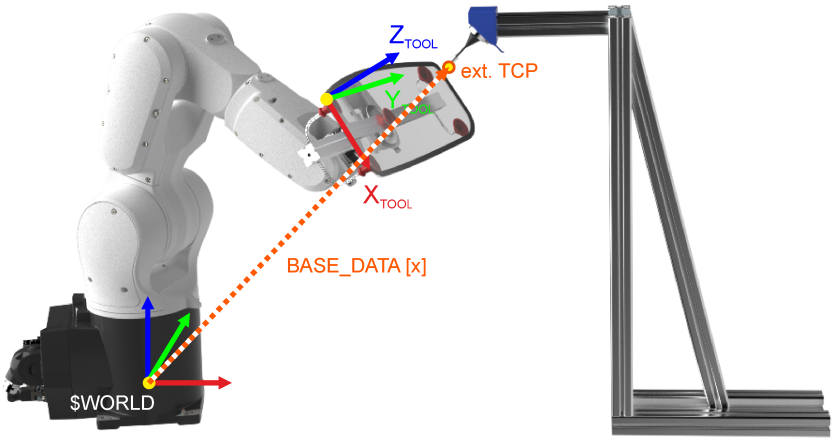


图 11-28：固定工具时的坐标系

11.4.1 练习：用外部 TCP 进行运动编程

练习内容

打开课程练习本并完成以下练习：

- 用外部 TCP 进行运动编程

12 专家界面介绍

12.1 学习单元：专家界面介绍

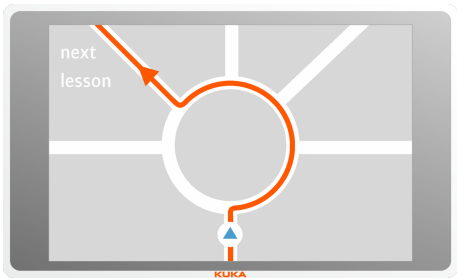


图 12-1：导航器

本学习单元讲授以下内容：

- 借助模板创建程序
- 调整资源管理器中的视图
- 调整编辑器中的视图
- 找到并排除程序中的错误
- 结构化机器人程序

12.2 借助模板创建程序

说明

- 在专家用户组中新建程序时有模板可供使用。
- 模板是用作新程序创建模板的程序框架（“样机”）。
- 原则上有以下模板可用。

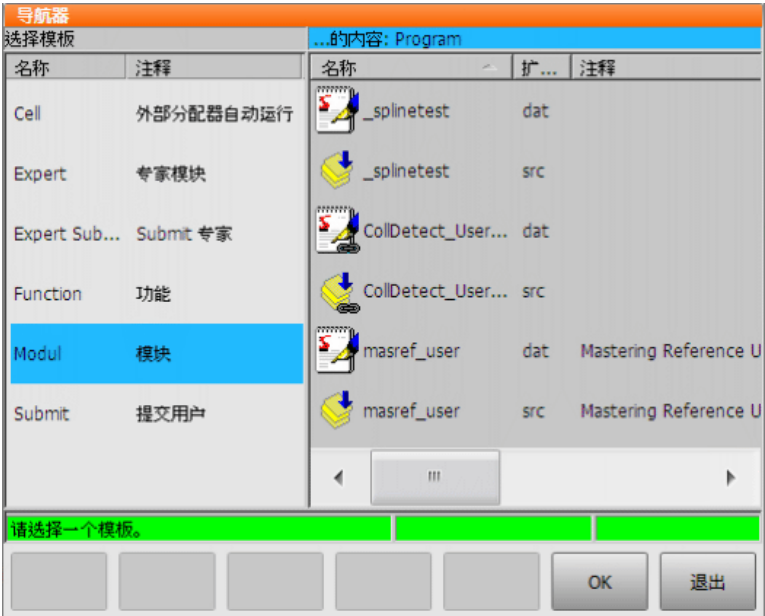


图 12-2：选择模板

- **Cell**: 现有的 Cell 程序，只能被替换或者在删除 Cell 程序后重新创建。
- **Expert**: 模块由只有程序头和程序结尾的 SRC 和 DAT 文件构成。

- **Expert Submit:** 附加的提交文件 (SUB) 由程序头和程序结尾构成。
- **Function:** SRC 函数创建, 在 SRC 中只创建带有 BOOL 变量的函数头。函数结尾已经存在, 但必须对返回值进行编程。
- **模块:** 模块由具有程序头、程序结尾以及基本框架 (INI 和 2 个 SPTP HOME) 的 SRC 和 DAT 文件构成。
- **Submit:** 附加的提交文件 (SUB) 由程序头、程序结尾以及基本框架 (DECLARATION、INI、LOOP/ENDLOOP) 构成。

修改模板

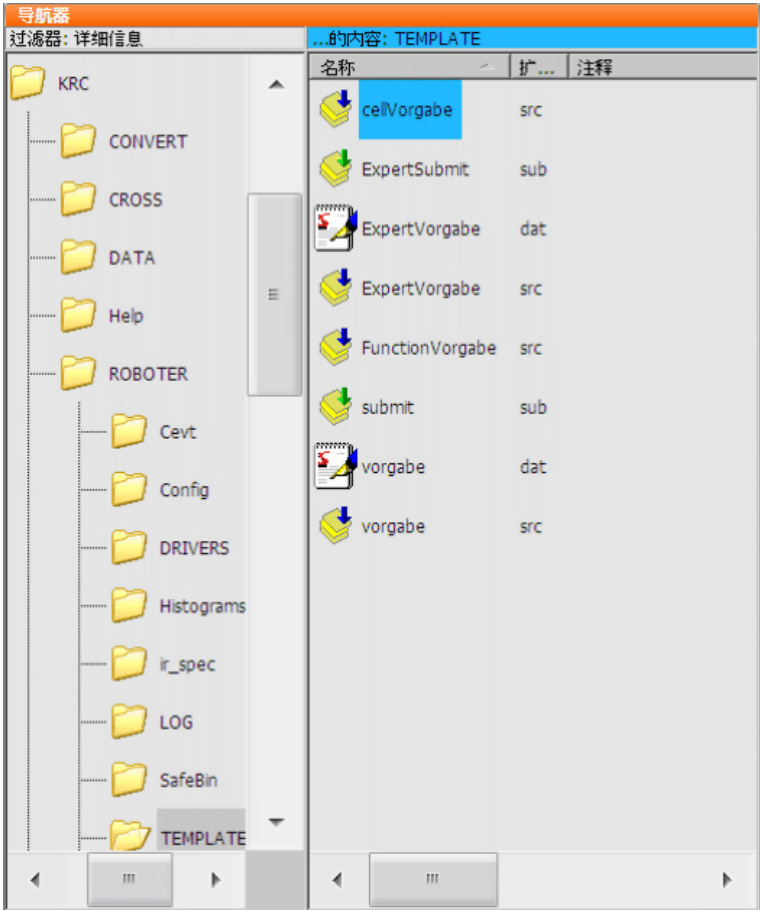


图 12-3: Templates



预设 ≡ 预定义

- 创建程序时可选的模板请见控制系统的以下
文件路径: `C:\KRC\ROBOTER\TEMPLATE`
- 模板是用作复制模板的预定义程序框架。
- 可以和“标准”程序、功能等一样对模板进行修改。

提示

如果要更改模板, 则该更改将立即影响程序的每次新建! 对已创建的程序全无影响。

12.3 调整浏览器中的视图

说明

使用详情视图

在专家用户组中，除了模块视图之外还可使用详情视图。

- 模块视图

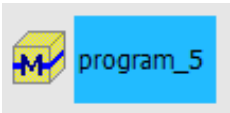


图 12-4：模块视图

- 在模块视图中始终将程序显示为条目。
- 实际的程序 *.src 和相关的列表 *.dat 可单独编辑。



如果复制模块，则始终一起复制所有所需的文件。

- 详细视图

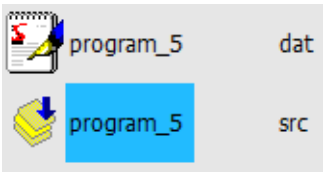


图 12-5：详细视图

- 在详情视图中分开显示实际的程序 *.src 和相关的列表 *.dat。
- 两个文件可相互独立地直接编辑、复制、删除和存档。



如果复制模块，则必须注意始终一起复制两个文件 *.src 和 *.dat。

操作步骤

- 通过编辑按钮打开弹出菜单。调用筛选器功能。



图 12-6：编辑，筛选器

- 在显示的选择窗口中可以在模块和详情视图之间切换。

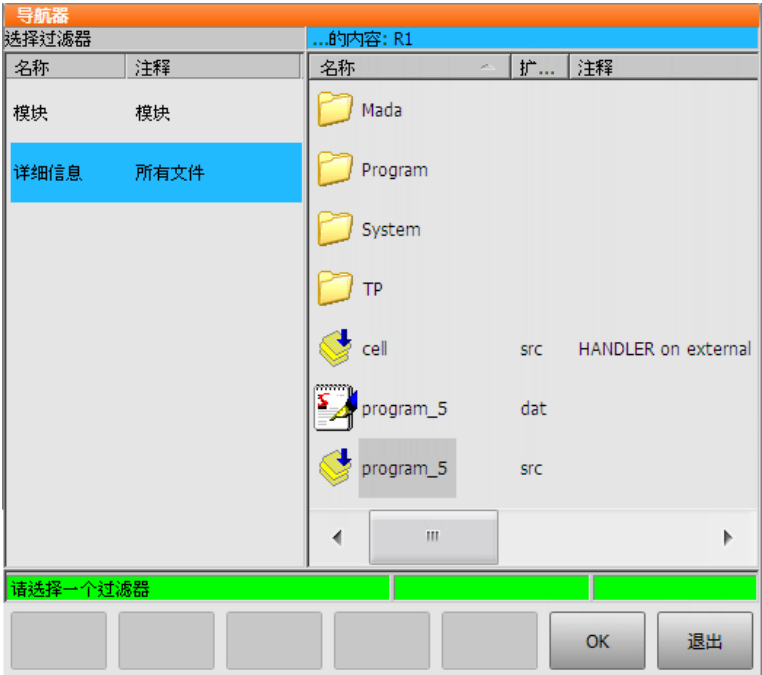


图 12-7：选择过滤器

12.4 调整编辑器中的视图

说明

- 可以在操作和显示方面调整编辑器中的视图。
- 即使细节深度也可以根据用户的需求进行调整。

操作步骤

1. 在 smartPAD 编辑器中打开一个程序。



图 12-8：视图

菜单路径：编辑按键 > 视图

- DEF 行
显示程序框架（程序头和 END）。

```
DEF program()  
...  
END
```

- **详细显示 (ASCII)**
显示隐藏的 ASCII 字符。

```
...  
;FOLD SPTP HOME VEL=100% DEFAULT;%{PE}%NJZJATOBASUS...  
...
```

- **换行**
在默认设置中，屏幕可水平滚屏。
如果换行已激活，则可以选择性地换行显示更长的程序行。
- **普通/小行间距**
程序行之间的行距更大或更小。

2. **示例：**编辑器中的扩展视图，带 DEF 行、详情显示和小换行。

```
编辑器  
1 DEF program_5( )  
2 ;FOLD INI;%{PE}  
3  
4 LOOP  
5  
6 ;FOLD SPTP HOME Vel=100 % DEFAULT;%{PE}%MKUKATPBASIS  
7 ;%CSPLINE,%USPTP_SB,%P 1:SPTP_SB, 2:HOME, 3:, 5:100,  
8 7:DEFAULT  
9  
10 ;FOLD SLIN P1 Vel=0.5 m/s CPDAT2 Tool[1]:Greifer  
11 Base[1]:Basis_Tisch ;%{PE}  
12 ;FOLD SLIN P2 Vel=0.5 m/s CPDAT3 Tool[1]:Greifer  
13 Base[1]:Basis_Tisch ;%{PE}  
14 ;FOLD SLIN P3 Vel=0.5 m/s CPDAT4 Tool[1]:Greifer  
15 Base[1]:Basis_Tisch ;%{PE}  
16  
17 ;FOLD SPTP HOME Vel=100 % DEFAULT ;%{PE}  
18  
19 ENDLLOOP  
20  
21 END  
22 |
```

图 12-9：详情视图，编辑器



高级视图（显示 DEF 行、打开细节视图）有助于根据行号查找句法错误。
(>>> [12.5 “找到并排除程序中的错误” 页面 339](#))
此外，所有可能存在的折叠夹都必须已打开。

12.5 找到并排除程序中的错误

说明

在用行指令创建程序时很少会产生编程错误。但是，如果用高级语言 KRL-KUKA Robot Language 进行编程，则在输入文本时可能导致句法错误。对于控制系统来说，这些无法解读，将作为错误显示。

- 错误的程序/文件在文件浏览器中以红色 X 标记。



图 12-10：有错误的程序

- 错误的程序/文件的上级文件夹同样用一个小红 X 标记。

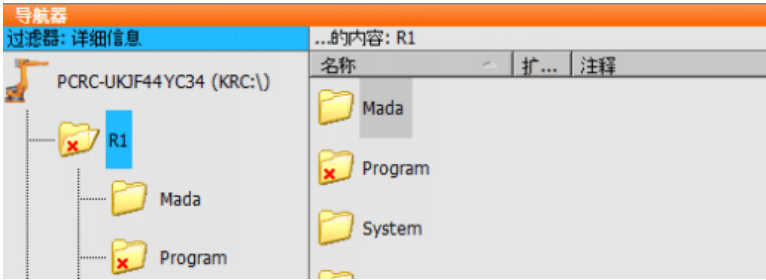


图 12-11：文件夹中的错误标记

操作步骤

- 1. 在导航器中选择出错的模块
(>>> 图 12-10)
- 2. 在
错误列表按钮上点击。



图 12-12：错误列表，调用

- 3. 错误显示（程序名.ERR）打开。

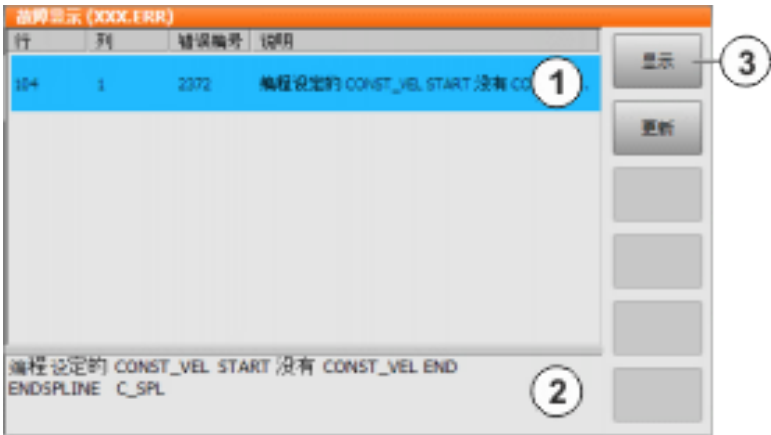


图 12-13：故障显示

- 在错误显示中以表格形式显示识别到的句法错误。
- 用鼠标选中行（1）。该行的背景色被置为蓝色。
- 在独立的窗口中显示详细的信息（2）。
- 该程序通过按键显示（3）打开。
- 4. 光标跳到导致错误的句法行中。该错误可以分析和纠正。

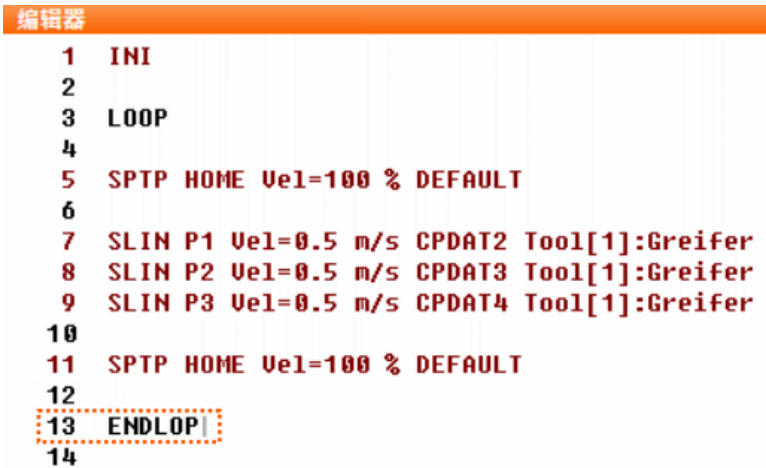


图 12-14：无限循环句法错误，“ENDLOOP”



在该示例中，在错误显示中显示的行编号 46 与程序 13 中的编号不一致。这是因为查看是在减小的显示中进行。（>>> [12.4 “调整编辑器中的视图” 页面 338](#)）

5. 退出编辑器并保存

12.6 结构化机器人程序

机器人程序结构化设计的方法

- 机器人程序的结构是体现其使用价值的一个十分重要的因数。
- 程序结构化越规范，程序就越易于理解、执行效果越好、越便于读取、越经济。
- 为了使程序得到结构化设计，可以使用以下技巧：
 - **添加注释**
注释和印记
(>>> [12.6.1 “给程序添加注释” 页面 341](#))
 - **缩进**
空格符
(>>> [12.6.2 “缩进程序” 页面 344](#))
 - **隐藏**
Fold （“折叠”）
(>>> [12.6.3 “隐藏程序行” 页面 344](#))
 - **模块技术**
使用子程序（模块）
(>>> [12.6.4 “链接机器人程序” 页面 345](#))

12.6.1 给程序添加注释

说明

添加注释可以在机器人程序里存储针对程序读者的文本。机器人解释器不会读取注释文本，注释只是使程序更易读。

在机器人程序中可以在许多地方使用注释：

- 程序文本**信息**：作者、版本、创建日期

```
Editor
1 DEF welding1( )
2 ; Programmed by JACK SPARROW
3 ; Version 1.5 (10/10/2010)
4 INI
5
6 PTP HOME Vel= 100 % DEFAULT
7
```

图 12-15: 注释示例: 信息

- 程序文本的**分段**: 主要在使用画图符号的情况下 (特殊字符 #、*、~)

```
0
7 ;*****Initialisation*****
8 INIT
9 BASISTECH INI
10 CHECK HOME
```

图 12-16: 注释示例: 分层目录

- **添加注释** (专家层面): 通过在程序行的起始位置添加分号来使该程序行“添加注释”, 即该文本作为注释来识别, 而不作为程序执行。

```
10 CHECK HOME
11 PTP HOME Vel= 100 % DEFAULT
12 ;$OUT[33]=TRUE
13 AUTOEXT INI
14 LOOP
```

图 12-17: 注释示例: 添加注释



行指令**无法**添加一个分号“;”。

- 对各个行的**解释**
以及对**需执行的工作的说明**: 标识未完成的程序段

```
17
18 CASE 1
19 P00 (#EXT_PGNO,#PGNO_ACKN,DMV[],0) ; Reset
↳ Progr.No.-Request
20 Main_Prog ( ) ; Call User-Program
21
```

图 12-18: 注释示例: 说明

提示

只有不断更新, 注释才有意义。如果之后更改过指令, 则也必须更新注释!

添加注释的种类

- **添加一个分号** (专家层面): 通过插入分号 (“;”) 使一行中的后面部分变成注释。

```
...
SPTP HOME Vel=100% DEFAULT

; ##### Wuerfel holen #####

SLIN P1 Vel=0.5 m/s CPDAT1 Tool[1]....
```

- 插入行指令“注释”



图 12-19：注释的联机表单

序号	说明
1	任意文字

- 插入行指令“印章”：
此外，还要在此插入一个时间戳记。此外，还可以插入编辑者的姓名。



图 12-20：印章的行指令

序号	说明
1	系统日期（可以更新）
2	系统时间（可以更新）
3	用户的名称或代码
4	任意文字

操作步骤

- 标记其后应插入注释或印章的那一行。
- 菜单顺序：命令 > 注释 > 标准或印章



图 12-21：插入注释

- 输入所希望的数据。

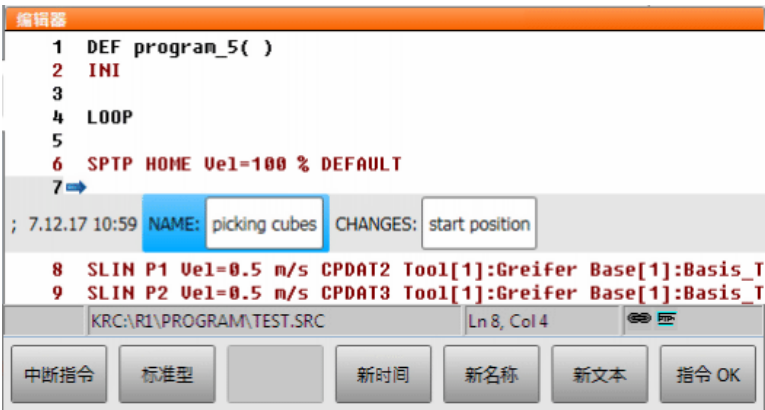


图 12-22：插入印章

- 插入注释时，可用**新文本**来清空注释栏，以便输入新的文字。
- 插入印章时，可用**新时间**来更新系统时间，并用**新名称**清空**名称**栏。



如果事先已经插入了注释或印章，则行指令中还保留着相同数据。

4. 用按键 **OK** 保存注释/印章。

12.6.2 缩进程序

程序行缩进

提高机器人程序可读性的一个有效手段是缩进程序行。由此可提高程序单元之间关联的清晰度。通过使用一个或多个空格来缩进。

```

13  AUTOEXT INI
14  LOOP
15      P00 (#EXT_PGNO,#PGNO_GET,DMY[],0 )
16      SWITCH PGNO ; Select with Programnumber
17          CASE 1
18              P00 (#EXT_PGNO,#PGNO_ACKN,DMY[],0 )
19              Main_Prog ( ) ; Call User-Program
20          CASE 2
21              P00 (#EXT_PGNO,#PGNO_ACKN,DMY[],0 )
22              Sub_Prog1 ( ) ; Call User-Program
23          DEFAULT
24              P00 (#EXT_PGNO,#PGNO_FAULT,DMY[],0 )
25      ENDSWITCH
26  ENLOOP

```

图 12-23：程序行缩进



缩进效应只是视觉上的。缩进的程序行在程序运行时与未缩进的程序行一样得到处理。

12.6.3 隐藏程序行

通过折叠隐藏程序行

KUKA 机器人编程语言可将程序行归纳和隐藏到**折叠**中。由此用户将无法看到这些程序行。提高读取程序的简便性。可在专家用户组中打开和关闭折叠。

```

13
14
15  CHECK HOME
16

```

图 12-24：关闭的 Fold

```

14
15  CHECK HOME
16  $H_POS=XHOME
17  IF CHECK_HOME==TRUE THEN
18      P00 (#CHK_HOME,#PGNO_GET,DMY[],0 ) ;Test HPos
19  ENDIF
20

```

图 12-25：打开的 Fold

折叠的颜色标记:

颜色	说明
深红	关闭的折叠
浅红	打开的折叠
深蓝	关闭的子折叠
浅蓝	打开的子折叠
绿色	Fold 内容

12.6.4 链接机器人程序

说明

利用子程序技术可将机器人程序模块化，因而可以有效地按结构设计程序。目的是不将所有指令写入一个程序，而是将特定的流程、计算或过程转移到单独的程序中。

通过使用子程序产生了一系列的优点：

- 由于程序长度减短，主程序结构更清晰并更易读。
- 子程序可以相互独立地开发：编程耗时可分摊，最小化错误源。
- 子程序可多次反复应用。
- 各个程序模块/子程序具有更好的可保养性和维护。

原则上，有两种不同的子程序类型：

- **全局子程序**
(>>> “全局子程序” 页面 345)
- **局部子程序**
(>>> “局部子程序” 页面 346)

全局子程序

一个全局子程序是一个独立的机器人程序（有自己的 *.SRC 和 *.DAT 文件），可由另一个机器人程序调用。可根据具体要求对程序进行分支，即某一程序可在某次应用中用作主程序，而在另一次则用作子程序。

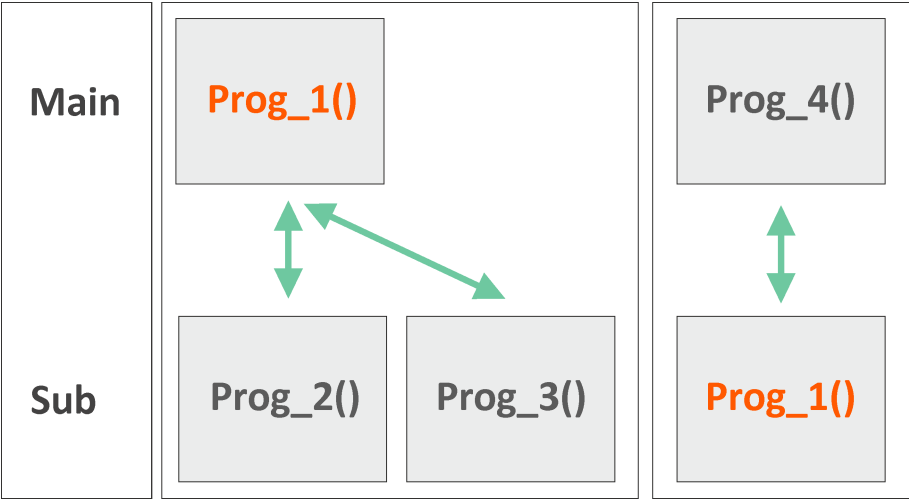


图 12-26：全局子程序示意图

局部子程序

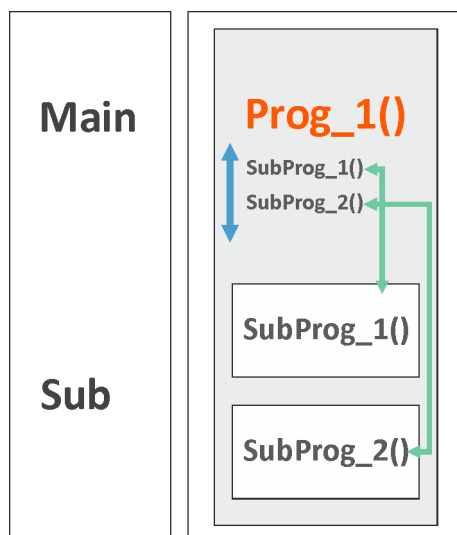


图 12-27：示意图：局部子程序

- 局部子程序是集成在一个主程序中的程序，即指令包含在同一个 SRC 文件中。
- 子程序的点坐标相应存放在同一个 DAT 文件中。

调用子程序的流程

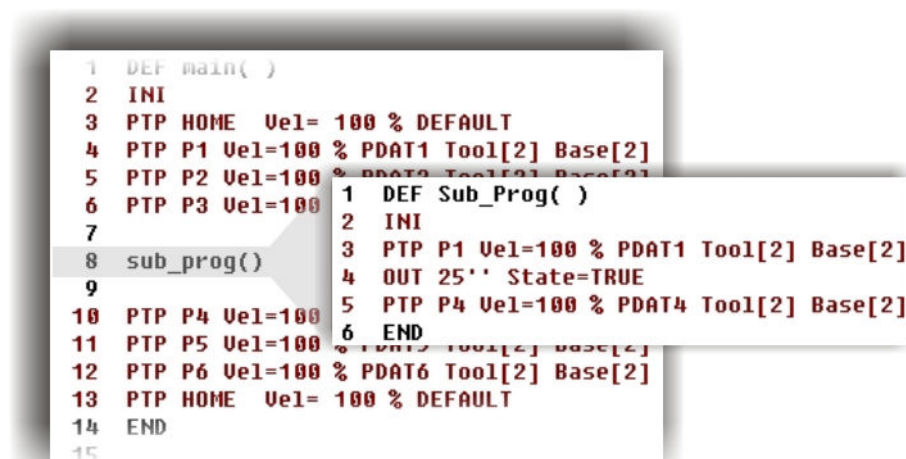


图 12-28：调用子程序的过程

- 每个程序都以 DEF 行开始并以 END 行结束。
- 如果要在主程序中调用子程序，则正常情况下子程序会从 DEF 运行至 END。
- 到达 END 行后，程序运行指针重新跳入发出调用指令的程序（主程序）。

提示

为了能提早退出子程序（即在 END 行之前），可以在子程序中编入 RETURN 指令。执行该程序行时，即可提前中止子程序的运行。

调用子程序

- 需要专家用户组。
(>>> 4.2.1 “切换用户组” 页面 80)
- 句法：名称()

1. 用**打开**将所需的主程序载入编辑器。

```
INI
SPTP HOME Vel= 100% DEFAULT

SPTP HOME Vel= 100% DEFAULT
```

2. 将光标定位在所需的行内。
3. 输入子程序名称和括号，例如：**myprog()**。

```
INI
SPTP HOME Vel= 100% DEFAULT
myprog( )
SPTP HOME Vel= 100% DEFAULT
```

4. 用关闭图标关闭编辑器并保存修改。

子程序的应用

```
DEF MAIN()

INI

    GET_PEN()
    PAINT_PATH()
    PEN_BACK()
    GET_PLATE()
    GLUE_PLATE()
    PLATE_BACK()

END
```

12.6.5 练习：编程子程序调用

练习内容

打开课程练习本并完成以下练习：

- 子程序调用编程

13 变量和协定

13.1 学习单元：变量和协议

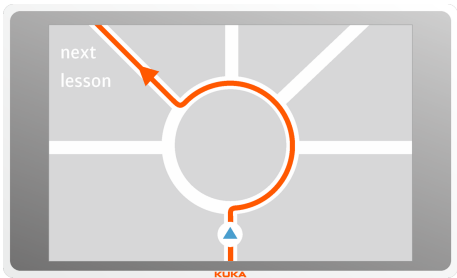


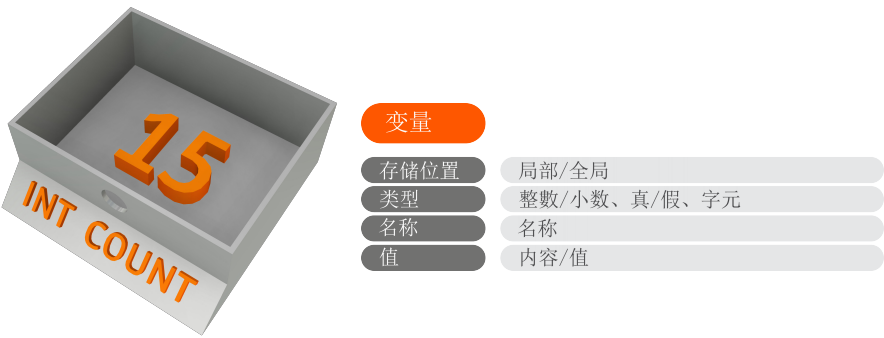
图 13-1：导航器

本学习单元讲授以下内容：

- KRL 中的数据管理
- 简单数据类型变量

13.2 KRL 中的数据管理

变量概述



- 在用 KRL 对机器人进行编程时，一个变量在通常的含义上就是机器人过程中出现的各种计算值（“值”）的容器。
- 每个变量都在计算机的存储器中有一个专门指定的地址。
- 一个变量用非 KUKA 关键词的名称来表示。
- 每个变量都属于一个专门的数据类型
- 在应用前必须声明数据类型。
- 在 KRL 中有局部（本地）变量和全局变量之分。

13.2.1 命名约定

在选择变量名称时，务必遵守以下规定：

- KRL 中的名称长度最多允许 24 个字符。
- KRL 中的名称允许含有字母（A - Z）、数字（0 - 9）以及特殊字符“_”和“\$”。
- KRL 中的名称不允许以数字开头。
- KRL 中的名称不允许为关键词。
- 大小写无关紧要。

建议

- 使用可让人一目了然的合理变量名称，
- 切勿使用缩写，
- 使用合理的名称长度，即不要每次都使用 24 个字符。

13.2.2 数据类型

预定义的默认数据类型

- **BOOL**: 经典式“是”/“否”结果
- **REAL**: 浮点数，为了避免四舍五入出错的运算结果
- **INT**: 整数，用于计数循环或件数计数器的经典计数变量
- **CHAR**: 仅一个字符
字符串或文本只能作为 CHAR 数组来实现

简单的数据类型	整数	浮点数	布尔数	单个字符
关键词	INT	REAL	BOOL	CHAR
值域	$-2^{31} \dots (2^{31}-1)$	$\pm 1.1 \cdot 10^{-38} \dots \pm 3.4 \cdot 10^{+38}$	TRUE / FALSE	ASCII 字符集
示例	-199 或 56	-0.0000123 或 3.1415	TRUE 或 FALSE	"A" 或 "q" 或 "7"

展望

基于标准数据类型还推导出了其他更复杂的数据类型，这些将在一门高级课程中用到。

数组 /Array

借助下标保存相同数据类型的多个变量。

```
Voltage[10] = 12.75
Voltage[11] = 15.59
```

枚举类型

枚举类型的所有值在创建时会定义名称（明文）。

```
color = #red
```

复合数据类型/结构体

结构体是由不同数据类型的数据项组成的一个复合数据类型。

```
Date = {day 14, month 12, year 1996}
```

13.2.3 建立变量

变量声明

- 在使用前必须总是先进行声明。
- 每一个变量须划归一种数据类型。
- 命名时要遵守命名规范。
- 声明的关键词为 **DECL**。

- 对于四种简单的数据类型，关键词 **DECL** 可省略。
- 在预进时赋值。
- 可在不同的位置进行变量声明。这将影响变量的生存期和有效性。

13.2.4 变量的双重声明

说明

双重声明总是产生于使用同样的变量名称（字符串）时。

在 `*.SRC/*.DAT` 文件和 `$config.dat` 中使用相同的变量名称时，**不是**双重声明。在该情况下，只要从声明了变量的程序中写入变量，则局部变量优先于全局变量。

如果从另一个程序中写出变量，则写出全局变量。

13.2.5 变量的生存期和有效性

使用寿命

- 生存期是为变量预留存储位置的时间。
- 运行时间变量在退出程序或功能时将其存储位置重新空出。
- 而数据列表中的变量将当前（上一个）值永久性保存在其存储位置中。

适用性

- 局部声明的变量仅在其被声明的程序中可用并可见。
- 而全局变量则建立在一个中央（全局）数据列表中。
- 全局变量也可建立在一个局部数据列表中，声明时冠以关键词 **global**（全局）。

13.2.6 取决于存储位置的变量声明

*.SRC 文件中的变量

- 在 `*.SRC` 文件中创建的变量称为运行时间变量。
- 不能始终显示该变量。
- 该变量仅在被声明的程序中可用。因此，这些变量在执行程序时可供使用。
- 在到达程序的最后一行（END 行）时重新释放存储位置。

局部 *.DAT 文件中的变量

- 在相关 `*.SRC` 文件的程序运行时始终可以被显示，
- 变量值在程序运行结束后还保持不变，
- 在整个 `*.SRC` 文件以及局部子程序中可用，
- 也可创建为全局变量，
- 将当前值保存在 `*.DAT` 文件中，重新调用时以所保存的值开始。
- 如果对该变量进行全局声明，则此变量也可供全局使用。如果 `DAT` 文件使用关键词 **PUBLIC** 并在声明时再另外使用关键词 **GOLBAL**，则该文件在所有程序中都可以读写。

系统文件 \$CONFIG.DAT 中的变量

- 可用于所有程序中（全局），
- 即使没有激活程序，也始终可以显示，
- 全局可用，即在所有程序中都可读写，

- 将当前值保存在 `$CONFIG.DAT` 中。

13.2.7 KUKA 系统数据

说明

- KUKA 统数据可能以所有数据类型出现，例如：
 - 枚举数据类型，例如：运行方式
 - 结构，例如：日期/时间
- 系统信息可从 KUKA 系统变量获得。这些变量
 - 读出当前系统信息，
 - 改变当前系统配置，
 - 已经预定义好并以 “\$” 字符开头
 - `$DATE` （当前时间和日期）
 - `$POS_ACT` （当前机器人位置）
 - `$MODE_OP` （当前运行方式）
 - ...

13.3 简单数据类型变量

在下文中将介绍如何创建、初始化和改变变量。在此仅使用简单的数据类型。

使用 KRL 的简单数据类型

- 整数 (INT)
- 实数 (REAL)
- 布尔值 (BOOL)
- 单个字符 (CHAR)

13.3.1 借助行指令对逻辑指令和循环编程

说明

KSS 8.6 以上，针对通过 smartPAD 的逻辑编程补充了其他行指令。



图 13-2：通过行指令对逻辑指令编程

这些行指令可简化编程，防止出现句法错误。行指令后面藏的是我们熟悉的 KRL 句法，当然它没有改变。对于更为复杂的结构编程，例如循环嵌套，建议仍然采用原本的 KRL 编程。



只能在 SRC 文件中插入行指令。为此必须和句法编程一样打开程序。

13.3.2 变量声明

变量声明的原理

SRC 文件中的程序结构范例

- 在声明部分必须声明变量
- 初始化部分从第一个赋值开始，但通常都是从“INI”行开始
- 在指令部分会赋值或更改值

```
DEF main( )  
; 声明部分  
...  
; 初始化部分  
INI  
...  
; 指令部分  
SPTP HOME Vel=100% DEFAULT  
...  
END
```

更改标准界面

- 只有作为专家才能使 DEF 行显示
- 为了在使用某些模块时于“INI”行前进入声明部分，该过程是必要的
- 为了能够看到 DEF 和 END 行，在将变量传递到子程序中时也非常重要

计划变量声明

- 规定生存期
- 规定有效性/可用性
- 规定数据类型
- 命名和声明

声明简单数据类型变量时的操作步骤**在 SRC 文件中创建变量**

1. 专家用户组
2. 使 DEF 行显示出来
3. 在编辑器中打开 SRC 文件
4. 声明变量

```
DEF MY_PROG ( )
DECL INT counter
DECL REAL price
DECL BOOL error
DECL CHAR symbol
INI
...
END
```

5. 关闭并保存程序

在 DAT 文件中创建变量

1. 专家用户组
2. 在编辑器中打开 DAT 文件
3. 声明变量

```
DEFDAT MY_PROG
EXTERNAL DECLARATIONS
DECL INT counter1
DECL REAL price1
DECL BOOL error1
DECL CHAR symbol1
...
ENDDAT
```

提示

不使用在 SRC 文件中使用的变量。这将导致双重声明。

4. 关闭并保存数据列表

在 \$CONFIG.DAT 中创建变量

1. 专家用户组
2. 在编辑器中打开 SYSTEM（系统）文件夹中的 \$CONFIG.DAT

```
DEFDAT $CONFIG
BASISTECH GLOBALS
AUTOEXT GLOBALS
USER GLOBALS
BackupManagerConfig
Conveyor
ENDDAT
```

3. 选择折叠 “USER GLOBALS”，然后用“打开/关闭折叠”按钮将其打开
4. 声明变量

```

DEFDAT $CONFIG
...
;=====
; Userdefined Types
;=====
;=====
; Userdefined Externals
;=====
;=====
; Userdefined Variables
;=====
DECL INT counter
DECL REAL price
DECL BOOL error
DECL CHAR symbol
...
; *****
; Make your modifications -ONLY- here
; *****
ENDDAT

```



\$CONFIG.DAT 可在软件更新期间和通过工艺程序包进行修改。只有单独标记的部分 “*Make your modifications -ONLY- here*” 例外。

5. 关闭并保存数据列表
- 在 DAT 文件中创建全局变量

1. 专用户组
2. 在编辑器中打开 DAT 文件
3. 通过关键词 PUBLIC 扩展程序头中的数据列表

```
DEFDAT MY_PROG PUBLIC
```

4. 声明变量

```

DEFDAT MY_PROG PUBLIC
EXTERNAL DECLARATIONS
DECL GLOBAL INT counter
DECL GLOBAL REAL price
DECL GLOBAL BOOL error
DECL GLOBAL CHAR symbol
...
ENDDAT

```

提示

不使用在 CONFIG.DAT 中使用的名称，这将导致双重声明。

5. 关闭并保存数据列表

13.3.3 简单数据类型变量的初始化

用 KRL 初始化的说明

- 在声明后变量都只预留了一个存储位置，其值总是无效值。
- 在 SRC 文件中声明和初始化始终在两个独立的行中进行。
- 在 DAT 文件中声明和初始化始终在一行中进行（只要需要初始化）。
常量仅可在数据表中声明且必须当时立即初始化。
如果上次分配的变量内容在数据列表中“可见”，则在此也必须进行“首次初始化”。
- 初始化部分在 *.src 文件中以第一次赋值开始。

初始化的方法

整数的初始化

- 初始化为十进制数

```
value = 58
```

- 初始化为二进制数

```
value = 'B111010'
```

计算：1*32+1*16+1*8+0*4+1*2+0*1 = 58

二进制	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰
十二月	32	16	8	4	2	1

- 初始化为十六进制数

```
value = 'H3A'
```

十六进制	Digit		计算		总和
A	10	*	16 ⁰	=	10
3	3	*	16 ¹	=	48
计算：3*16 + 10 = 58					58

十六进制	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
十二月	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

使用 KRL 初始化时的操作步骤

在 SRC 文件中声明和初始化

- 在编辑器中打开 SRC 文件
- 已声明完毕
- 执行初始化

```
DEF MY_PROG ( )
DECL INT counter
DECL REAL price
DECL BOOL error
DECL CHAR symbol
INI
```

```

counter = 10
price = 0.0
error = FALSE
symbol = "X"
...
END

```

4. 关闭并保存程序

在 DAT 文件中声明和初始化

1. 在编辑器中打开 DAT 文件
2. 已声明完毕
3. 执行初始化

```

DEFDAT MY_PROG
  EXTERNAL DECLARATIONS
  DECL INT counter = 10
  DECL REAL price = 0.0
  DECL BOOL error = FALSE
  DECL CHAR symbol = "X"
  ...
ENDDAT

```

4. 关闭并保存数据列表

在 DAT 文件中声明和在 SRC 文件中初始化

1. 在编辑器中打开 DAT 文件
2. 进行声明

```

DEFDAT MY_PROG
  EXTERNAL DECLARATIONS
  DECL INT counter
  DECL REAL price
  DECL BOOL error
  DECL CHAR symbol
  ...
ENDDAT

```

3. 关闭并保存数据列表
4. 在编辑器中打开 SRC 文件
5. 执行初始化

```

DEF MY_PROG ( )
  ...
  INI
  counter = 10
  price = 0.0
  error = FALSE
  symbol = "X"
  ...
END

```

6. 关闭并保存程序

常量的声明和初始化

说明

- 常量用关键词 **CONST** 建立。

- 常量只允许在数据列表中建立。

创建常量

1. 在编辑器中打开 DAT 文件。
2. 进行声明和初始化。

```
DEFDAT MY_PROG
EXTERNAL DECLARATIONS
DECL CONST INT max_size = 99
DECL CONST REAL PI = 3.1415
...
ENDDAT
```

3. 关闭并保存数据列表。

13.3.4 通过行指令声明并初始化变量



图 13-3：通过行指令对逻辑指令编程

1. 在程序中选中要插入行指令的位置
2. 点击指令按钮（1）和逻辑按钮（2）。
3. 在选项窗口中选择所需逻辑指令（3），示例中为变量声明。
4. 调整插入的行指令，并根据函数利用输入按钮扩展行指令。

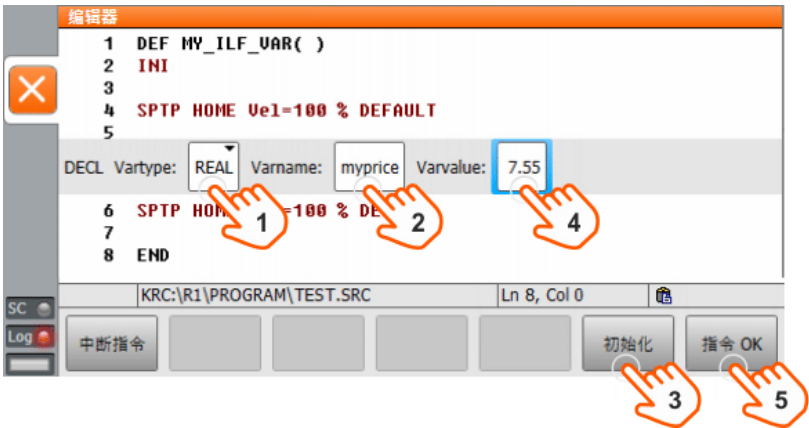


图 13-4：行指令中的变量、声明和初始化

在我们的示例“变量声明”中选择数据类型（Vartype）（1），并确定变量名称（Varname）（2）。如果要同时将变量首次初始化，则点击初始化按钮（3）扩展行指令。在 Varvalue 栏（4）中输入初始化值，并点击指令 OK 按钮（5）保存输入内容。

5. 变量声明和初始化应用到现有程序中。初始化位于**折叠：INI > User INI** 中。

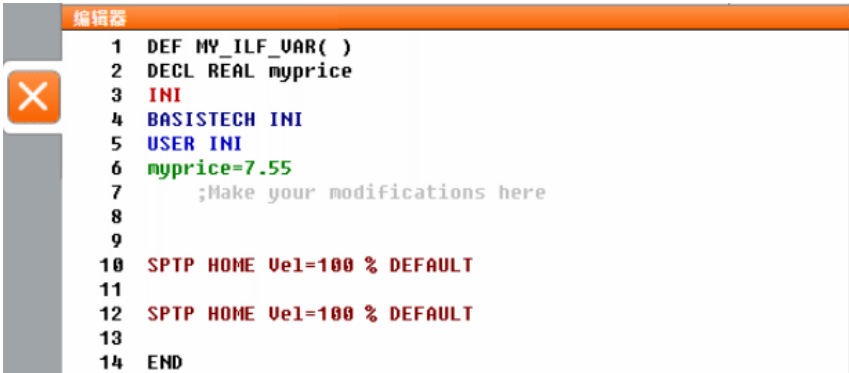


图 13-5：应用到 KRL 中

“更改”时的限制

通过按键 **更改** 无法更改或添加初始化。
可能的更改：更改变量的名称和类型

13.3.5 用 KRL 对简单数据类型的变量值进行操纵

用 KRL 修改变量值的方法列表

根据具体任务，可以以不同方式在程序进程（SRC 文件）中改变变量值。以下介绍最常用的方法。也可借助于位运算和标准函数进行操作，但不在在此深入介绍。

数据操作借助

- 基本运算类型
 - (+) 加法
 - (-) 减法
 - (*) 乘法
 - (/) 除法
- 比较运算

- (==) 相同 / 等于
- (<>) 不同
- (>) 大于
- (<) 小于
- (>=) 大于等于
- (<=) 小于等于
- 逻辑运算
 - (NOT) 取反
 - (AND) 逻辑 “与”
 - (OR) 逻辑 “或”
 - (EXOR) “异或”
- 位运算
 - (B_NOT) 按位取反运算
 - (B_AND) 按位与
 - (B_OR) 按位或
 - (B_EXOR) 按位异或

标准函数

- 绝对值函数
- 平方根函数
- 正弦和余弦函数
- 正切函数
- 反余弦函数
- 反正切函数
- 多种字符串处理函数

数据操作时的关系

使用数据类型 REAL 和 INT 时的数值更改

- 四舍五入

```

; 声明
DECL INT A,B,C
DECL REAL R,S,T
; 初始化
A = 3           ; A=3
B = 5.5         ; B=6 (X.5 起入为整数)
C = 2.25        ; C=2 (取整数)
R = 4           ; R=4.0
S = 6.5         ; S=6.5
T = C           ; T=2.0 (应用四舍五入后的整数)
```



经验法则:
步骤 1: 计算中间结果 (= 右侧)。
步骤 2: 然后进行赋值, 赋值时必须根据目标数据类型进行四舍五入 (INT 用作目标数据类型)

- 数学运算结果 (+;-;*)

结果	运算数 1	操作	运算数 2
INT	INT	+/-/*	INT
REAL	INT	+/-/*	REAL
REAL	REAL	+/-/*	INT



结果	运算数 1	操作	运算数 2
REAL	REAL	+/-/*	REAL

经验法则：
只要运算数具有 REAL 数据类型，则得出的结果也为 REAL。

```
; 声明
DECL INT D,E
DECL REAL U,V
; 初始化
D = 2
E = 5
U = 0.5
V = 10.6
; 指令部分（数据操作）
D = D*E ; D = 2 * 5 = 10
E = E+V ; E= 5 + 10.6 = 15.6 -> 取整数 E=16
U = U*V ; U= 0.5 * 10.6 = 5.3
V = E+V ; V= 16 + 10.6 = 26.6
```

- 数学运算结果 (/)
- 使用整数值运算时的特点：
- 纯整数运算的中间结果将去掉所有小数位。
 - 给整数变量赋值时将根据一般计算规则对结果进行四舍五入。

```
; 声明
DECL INT F
DECL REAL W
; 初始化
F = 10
W = 10.0
; 指令部分（数据操作）
; INT / INT -> INT
F = F/2 ; F=5
F = 10/4 ; F=2 (10/4 = 2.5 -> 去掉小数位)
; REAL / INT -> REAL
F = W/4 ; F=3 (10.0/4=2.5 -> 取整数)
W = W/4 ; W=2.5
```

比较运算

通过比较运算可以构成逻辑表达式。比较结果始终是布尔（BOOL）数据类型。

运算符/KRL	说明	允许的数据类型
==	等于/相等	INT、REAL、CHAR、BOOL
<>	不等	INT、REAL、CHAR、BOOL
>	大于	INT、REAL、CHAR
<	小于	INT、REAL、CHAR
>=	大于等于	INT、REAL、CHAR
<=	小于等于	INT、REAL、CHAR

```
; 声明
DECL BOOL G,H
```



```

; 初始化/指令部分
G = 10>10.1 ; G=FALSE
H = 10/3 == 3 ; H=TRUE
G = G<>H ; G=TRUE

```

逻辑运算

通过逻辑运算可以构成逻辑表达式。这种运算的结果始终是布尔（BOOL）数据类型。

运算		NOT A	A AND B	A OR B	A EXOR B
A=FALSE	B=FALSE	TRUE	FALSE	FALSE	FALSE
A=FALSE	B=TRUE	TRUE	FALSE	TRUE	TRUE
A=TRUE	B=FALSE	FALSE	FALSE	TRUE	TRUE
A=TRUE	B=TRUE	FALSE	TRUE	TRUE	FALSE

```

; 声明
DECL BOOL K, L, M
; 初始化/指令部分
K = TRUE
L = NOT K ; L=FALSE
M = (K AND L) OR (K EXOR L) ; M=TRUE
L = NOT (NOT K) ; L=TRUE

```

运算将根据其优先级顺序进行

优先级	运算符
1	NOT (B_NOT)
2	乘 (*); 除 (/)
3	加 (+), 减 (-)
4	AND (B_AND)
5	EXOR (B_EXOR)
6	OR (B_OR)
7	各种比较 (==; <>; ...)

```

; 声明
DECL BOOL X, Y
DECL INT Z
; 初始化/指令部分
X = TRUE
Z = 4
Y = (4*Z+16 <> 32) AND X ; Y=FALSE

```

数据操纵时的操作步骤

1. 确定一个或多个变量的数据类型
2. 确定变量的有效性和生存期
3. 进行变量声明
4. 初始化变量
5. 在程序运行中，即始终在 *.SRC 文件中对变量进行操纵
6. 关闭并保存 *.SRC 文件

HALT（停止）指令

停止指令主要在编程阶段用于测试。 例如：为了将运行时间变量发送到显示屏上。

- 停止指令使程序停止。 但是最后一次进行的运动指令仍然完整执行。
- 程序仅可用启动键继续进行。 随后执行“停止”之后的下一个指令。



中断程序中，程序在预进过程完整执行完毕后才被停止。

例如：

```
DEF program()
DECL BOOL a,b
INI
...
SPTP XP1
a=$IN[1]
b=$IN[2]
HALT
IF ((a == TRUE) AND (b == FALSE)) THEN
..
ENDIF
...
```

13.3.6 练习：简单的数据类型

练习内容

打开课程练习本并完成以下练习：

- 简单的数据类型

14 使用程序流程控制

14.1 学习单元：程序运行控制的用处

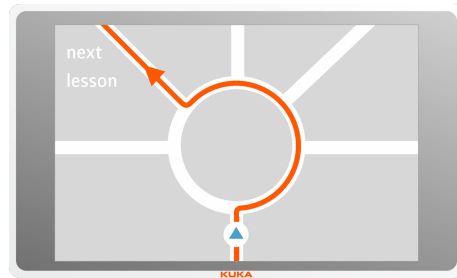


图 14-1：导航器

本学习单元讲授以下内容：

- 循环的编程
- 编程分支
- 多分支的编程
- 编程跳转指令
- 用 KRL 语言给等待功能编程

14.2 给循环编程

除了纯运动指令和通讯指令（切换和等待函数）之外，在机器人程序中还有大量用于控制程序流程的例程。

循环概述

- 循环是控制结构。
- 它不断重复执行程序指令，直至出现中断条件。
- 不允许从外部跳入循环结构中
- 循环可互相嵌套
- 有不同的循环类型
 - 无限循环
 - 计数循环
 - 条件循环
 - 当型循环
 - 直到型循环

14.2.1 Loop 循环编程

说明

Loop 循环是每次运行完之后都会重新运行的循环。运行过程可通过外部控制而终止。

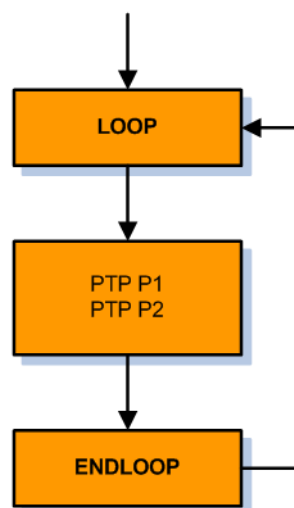


图 14-2：程序流程图：无限循环

- Loop 循环可直接用 EXIT 退出
- 用 EXIT 退出 Loop 循环时必须注意避免碰撞
- 如果两个 Loop 循环互相嵌套，则需要两个 EXIT 指令以退出两个循环

句法

```

LOOP
; 指令
...
; 指令
ENDLOOP

```

Loop 循环编程的示例

无中断的 Loop 循环

```

DEF MY_PROG( )
INI
SPTP HOME Vel=100% DEFAULT

LOOP
  SPTP P1
  SPTP P2
  SPTP P3
  SPTP P4
ENDLOOP

SPTP P5 Vel=30% PDAT5 Tool[1] Base[1]
SPTP HOME Vel=100% DEFAULT
END

```



因编程技术而从不驶至点 P5

带中断的 Loop 循环

```

DEF MY_PROG( )
INI
SPTP HOME Vel=100% DEFAULT

```

```
LOOP
  SPTP P1
  SPTP P2
; Bedingung für Abbruch
  IF $IN[3]==TRUE THEN ;
    EXIT
  ENDIF
  SPTP P3
  SPTP P4
ENDLOOP

SPTP P5 Vel=30% PDAT5 Tool[1] Base[1]
SPTP HOME Vel=100% DEFAULT
END
```



只要输入端 3 激活，则会驶至点 P5。
重要提示：对于 P2 和 P5 之间的运动必须检查是否会发生碰撞

通过行指令 Loop 循环

```
LOOP
```

图 14-3：不带栏的行指令 LOOP ... ENDLOOP

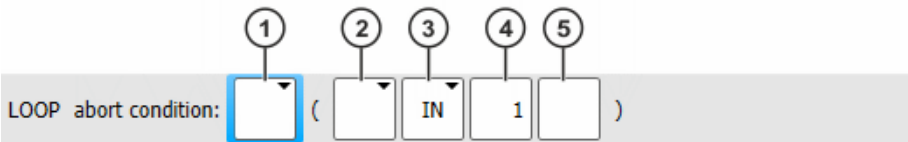


图 14-4：带栏的行指令 LOOP ... ENDLOOP

为了将栏显示出来，触碰按键 添加中断 条件。

序号	说明
1	• NOT: 添加 NOT。 • [空]
	添加括号外部逻辑运算。该项位于两个括号的表达式之间。 通过相应的按键添加所需的项： • AND、OR 或 EXOR
2	• NOT: 添加 NOT。 • [空]
	添加括号内部逻辑运算。运算符位于一个括号的表达式内。 通过相应的按键添加所需的运算符： • AND、OR 或 EXOR
3	要分析的信号。默认选择： • IN、OUT、CYCFLAG、TIMER 或 FLAG 也可以编程其他信号：替代默认选择，可以通过按键 自由文本 显示可在其中输入 KRL 的栏。 系统列表 重新显示默认选择。通过触碰按键可以在它们之间切换。
4	只在从默认选择中选择一个信号时，该栏才存在。 输入信号的编号。

序号	说明
5	只在从默认选择中选择一个信号时，该栏才存在。如果信号已有注释则会显示出来。 从专家用户组起：可以输入名称或更改现有的名称。只要还没有保存行指令，则可以通过点击 长文本 重新复位这些更改。

“更改”时的限制

无法通过按键 **更改** 更改行指令！

14.2.2 FOR 循环编程

说明

FOR 循环是一种可以通过规定重复次数执行一个或多个指令的控制结构。
要进行 FOR 循环则必须事先声明 Integer 数据类型的循环计数器。

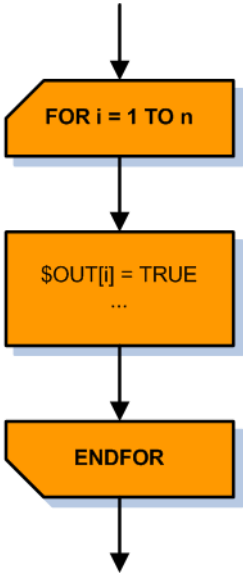


图 14-5：程序流程图： 计数循环



increment 的值必须作为整数输入并且可能不是变量。

标准步幅 +1 的句法

```
FOR counter = start TO last
;指令
ENDFOR
```

FOR 循环从值 start 开始，最迟于值 last 结束。

针对不同步幅的句法

```
FOR counter = start TO last STEP increment
;指令
ENDFOR
```

步幅 (increment) 也可通过关键词 STEP 指定为某个整数。
increment 的值必须作为整数输入并且**不能**是变量。

该 FOR 循环可借助 EXIT 立即退出。

工作原理

```
DECL INT counter

FOR counter = 1 TO 3 Step 1
;指令
ENDFOR
```

1. 用起始值将循环计数器初始化: counter = 1
2. 循环计数器在 ENDFOR 时会以步幅 STEP 递增计数
3. 循环又从 FOR 行开始
4. 检查进入循环的条件: 循环计数器必须小于等于指定的终值, 否则会结束循环
5. 根据检查结果的不同, 循环计数器会再次递增计数或结束循环。结束循环后程序在 ENDFOR 行后继续运行

没有指定步幅的单层 FOR 循环

```
DECL INT counter

FOR counter = 1 TO 50
    $OUT[counter] = FALSE
ENDFOR
```



没有借助 STEP 指定步幅时, 会自动使用步幅 +1。

指定步幅的单层 FOR 循环

```
DECL INT counter

FOR counter = 1 TO 4 STEP 2
    $OUT[counter] = TRUE
ENDFOR
```



该循环只会运行两次。第一次以起始数值 counter=1, 第二次则以 counter=3。计数值为 5 时, 循环立即中止。

倒数计数

指定负向步幅的 FOR 循环

```
DECL INT counter

FOR counter = 10 TO 1 STEP -1
    value = 100 - 2 * counter
ENDFOR
```



循环的初始值或者起始值必须大于等于终值, 以便循环能够多次运行。

嵌套 FOR 循环

指定步幅的嵌套 FOR 循环

```
DECL INT counter1, counter2

FOR counter1 = 1 TO 21 STEP 2
    FOR counter2 = 20 TO 2 STEP -2
```



```
...
ENDFOR
ENDFOR
```



每次都会先运行内部循环（此处以 counter2），然后运行外部循环（counter1）。

借助行指令对 FOR 循环编程

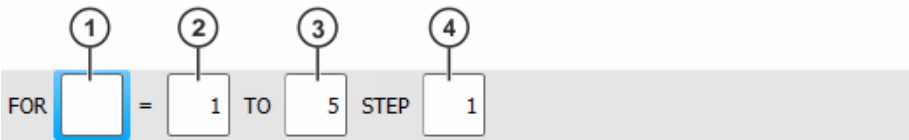


图 14-6：行指令 FOR ... ENDFOR

序号	说明
1	对循环进行计数的 INT 变量，“计数变量”。变量可能，但是不必事先声明。 该值可以在循环之内和之外的指令中使用。离开循环之后，该变量拥有最后接受的值。
2	起始值 自动给计数变量预分配起始值。
3	终值 在超出或低于终值时，循环结束。
4	步幅 每次循环执行结束后，计数变量自动以步幅变化。该值不得为负。默认：1。 • 正值：在计数变量大于终值时，循环结束。 • 负值：在计数变量小于终值时，循环结束。 该值不允许为零或变量。

14.2.3 WHILE 循环编程

说明

WHILE 循环也被称为前测试型循环。它是一种当型或者前测试型循环，这种循环会在执行循环的指令部分前先检查执行条件。

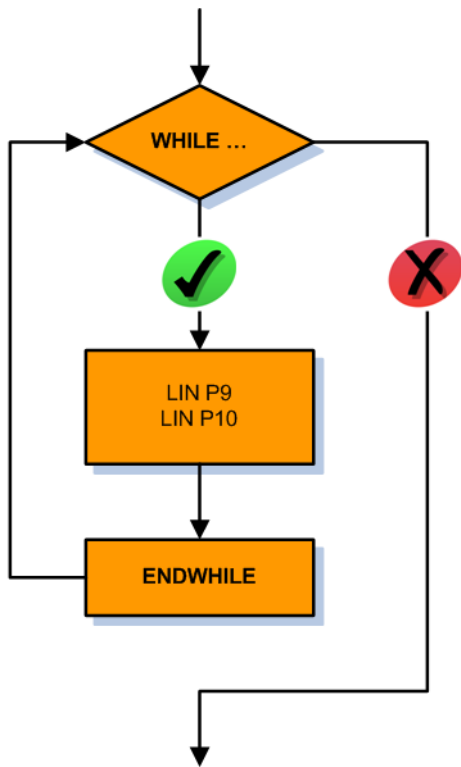


图 14-7：程序流程图： 当型循环

句法

只要某一执行条件（condition）得到满足，WHILE 循环就会一直将过程重复下去。执行条件不满足时会导致立即结束循环，并执行 ENDWHILE 后的指令

```
WHILE condition
; 指令
ENDWHILE
```

WHILE 循环可通过 EXIT 指令立即退出。

用一个 WHILE 循环编程

- 具有简单执行条件的 WHILE 循环

```
...
WHILE $IN[41]==TRUE ; 部件备好在库中
    PICK_PART ( )
ENDWHILE
...
```



表达式 WHILE \$IN[41]==TRUE 也可简化为 WHILE \$IN[41]。省略始终表示比较为真（TRUE）。

- 具有简单否定型执行条件的 WHILE 循环

```
...
WHILE $IN[41]==FALSE ; 库为空
    PICK_PART ( )
```

```
ENDWHILE...
```

或者

```
...  
WHILE NOT $IN[41]==TRUE ; 库为空  
    PICK_PART( )  
ENDWHILE...
```

• 具有复合执行条件的 WHILE 循环

```
...  
WHILE (( $IN[40]==TRUE) AND ( $IN[41]==FALSE) OR (counter>20))  
    PALLET( )  
ENDWHILE  
...
```

借助行指令对 WHILE 循环编程

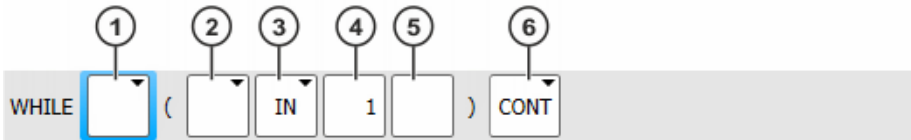


图 14-8：行指令 WHILE ... ENDWHILE

序号	说明
1	• NOT: 添加 NOT。 • [空]
	添加括号外部逻辑运算。该项位于两个括号的表达式之间。 通过相应的按键添加所需的项： • AND、OR 或 EXOR
2	• NOT: 添加 NOT。 • [空]
	添加括号内部逻辑运算。运算符位于一个括号的表达式内。 通过相应的按键添加所需的运算符： • AND、OR 或 EXOR
3	要分析的信号。默认选择： • IN、OUT、CYCFLAG、TIMER 或 FLAG 也可以编程其他信号。为此，有以下按键可用。触碰按键将显示下一个。 • 自由文本: 显示在其中可输入 KRL 的栏。 • 用户列表: 显示带自定义变量的列表。前提条件：配置了该列表。 • 系统列表: 重新显示默认选择。
4	• 如果从默认选择中选择一个信号：输入编号。 • 如果从用户列表选择一个变量：输入值。

序号	说明
5	只在从默认选择中选择一个信号时，该栏才存在。如果信号已有注释则会显示出来。 从专家用户组起：可以输入名称或更改现有的名称。只要还没有保存行指令，则可以通过点击 长文本 重新复位这些更改。
6	• CONT：指令被预进指针执行 • [空]：指令触发预进停止

“更改”时的限制

通过按键 **更改** 只能更改条件。CONT 无法添加或删除。

14.2.4 REPEAT...UNTIL 循环编程

说明

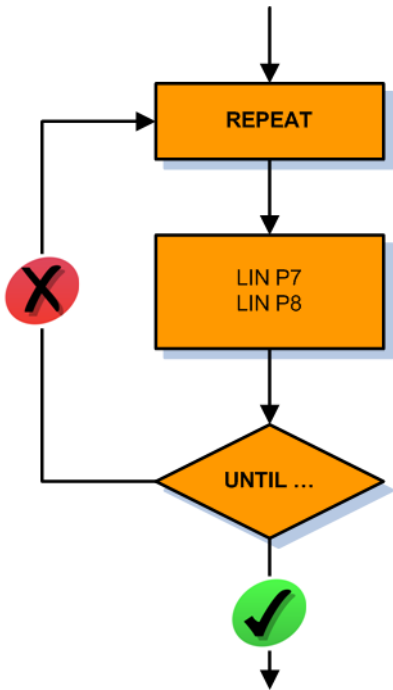


图 14-9：程序流程图： 直到型循环

- REPEAT...UNTIL 循环也称为后测试循环。
- REPEAT 循环是一种直到型或者后测试型循环，这种循环会在第一次执行完循环的指令部分后才会检测终止条件。
- 在指令部分执行完毕之后，检查是否已满足退出循环的条件 (condition)。
 - 条件满足时，退出循环，执行 UNTIL 后的指令。
 - 条件不满足时，在 REPEAT 处重新开始循环。
- 句法

```
REPEAT
; 指令
UNTIL condition
```

- REPEAT...UNTIL 循环可通过 EXIT 指令立即退出。

REPEAT...UNTIL 循环编程

- 具有简单执行条件的 REPEAT...UNTIL 循环编程。

```
...
REPEAT
  PICK_PART( )
UNTIL $IN[42]==TRUE ; 输入端 42: 库为空
...
```



表达式 UNTIL IN[42]==TRUE 也可简化为 UNTIL IN[42]。省略始终表示比较为真 (TRUE)

- 具有复杂执行条件的 REPEAT...UNTIL 循环。

```
...
REPEAT
  PALLET( )
UNTIL (($IN[40]==TRUE) AND ($IN[41]==FALSE) OR (counter>20))
...
```



结果为正时将结束循环！

借助行指令对 REPEAT...UNTIL 循环编程

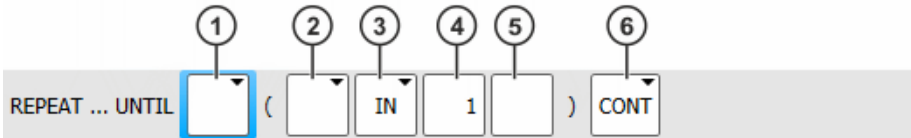


图 14-10: 行指令 REPEAT ... UNTIL

序号	说明
1	- NOT: 添加 NOT。 [空]
	添加括号外部逻辑运算。该项位于两个括号的表达式之间。 通过相应的按键添加所需的项: AND、OR 或 EXOR
2	- NOT: 添加 NOT。 [空]
	添加括号内部逻辑运算。运算符位于一个括号的表达式内。 通过相应的按键添加所需的运算符: AND、OR 或 EXOR
3	要分析的信号。默认选择: - IN、OUT、CYCFLAG、TIMER 或 FLAG 也可以编程其他信号。为此，有以下按键可用。触碰按键将显示下一个。 - 自由文本: 显示在其中可输入 KRL 的栏。 - 用户列表: 显示带自定义变量的列表。前提条件: 配置了该列表。 - 系统列表: 重新显示默认选择。

序号	说明
4	- 如果从默认选择中选择一个信号：输入编号。 - 如果从用户列表中选择一个变量：输入值。
5	只在从默认选择中选择一个信号时，该栏才存在。如果信号已有注释则会显示出来。 从专家用户组起：可以输入名称或更改现有的名称。只要还没有保存行指令，则可以通过点击 长文本 重新复位这些更改。
6	CONT：指令被预进指针执行 [空]：指令触发预进停止

“更改”时的限制

通过按键 **更改** 只能更改条件。CONT 无法添加或删除。

14.3 IF 语句编程

说明

使用 if 分支后，便可以只在特定的条件下执行程序段。它用于将程序分为多个路径。条件性分支（IF 语句）由一个条件和两个指令部分组成。

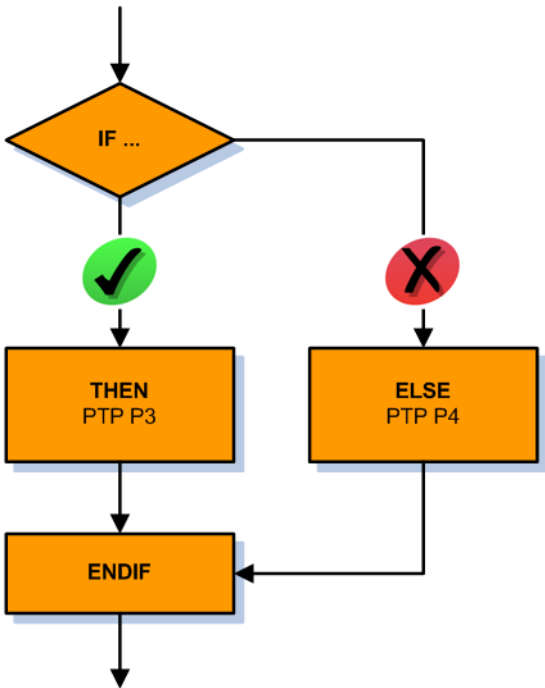


图 14-11：程序流程图： IF 分支

IF 指令检查此条件为真（TRUE）或为假（FALSE）。

- 如果满足条件，则可处理第一个指令。
- 如果未满足条件，则执行第二个替代指令。

IF 语句的类型

- 第二个指令部分可以省去：无 ELSE 的 IF 语句。由此，当不满足条件时紧跟在分支（ENDIF）后便继续执行程序。
- 多个 IF 语句可相互嵌套（多重分支）：语句被依次处理并且检查是否有一个条件得到满足。

为分支编程

句法

- 带选择分支语句

```
IF condition THEN  
  指令  
ELSE  
  指令  
ENDIF
```

- 无选择分支语句（询问）

```
IF condition THEN  
  指令  
ENDIF
```

if 分支示例

有可选分支的 if 分支

```
DEF MY_PROG( )  
DECL INT error_nr  
...  
INI  
error_nr = 4  
...  
; 仅在 error_nr = 5 时驶至 XP21 , 否则 XP22  
IF error_nr == 5 THEN  
  SPTP XP21 指令 1  
ELSE  
  SPTP XP22 指令 2  
ENDIF  
...  
END
```

没有可选分支的 if 分支

```
DEF MY_PROG( )  
DECL INT error_nr  
...  
INI  
error_nr = 4  
...  
; 仅在 error_nr = 5 时驶至 XP21  
IF error_nr == 5 THEN  
  SPTP XP21  
ENDIF  
...  
END
```

有复杂执行条件的 if 分支

```
DEF MY_PROG( )  
DECL INT error_nr  
...  
INI  
error_nr = 4  
...  
; 仅在 error_nr = 1 或 10 或大于 99 时驶至 XP21  
IF ((error_nr == 1) OR (error_nr == 10) OR (error_nr > 99)) THEN
```

```
SPTP XP21
ENDIF
...
END
```

有布尔表达式的 if 分支

```
DEF MY_PROG( )
DECL BOOL no_error
...
INI
no_error = TRUE
...
; 仅在无故障 (no_error) 时驶至 XP21
IF no_error == TRUE THEN
    SPTP XP21
ENDIF
...
END
```



表达式 IF no_error==TRUE THEN 也可以简化为 IF no_error THEN。省略始终表示比较为真（TRUE）。

用行指令对询问或分支编程

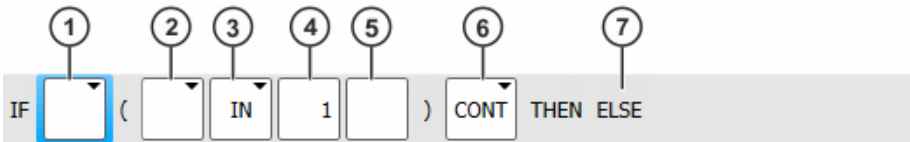


图 14-12：行指令 IF ... THEN

序号	说明
1	• NOT: 添加 NOT。 • [空]
	添加括号外部逻辑运算。该项位于两个括号的表达式之间。 通过相应的按键添加所需的项： • AND、OR 或 EXOR
2	• NOT: 添加 NOT。 • [空]
	添加括号内部逻辑运算。运算符位于一个括号的表达式内。 通过相应的按键添加所需的运算符： • AND、OR 或 EXOR
3	要分析的信号。默认选择： • IN、OUT、CYCFLAG、TIMER 或 FLAG 也可以编程其他信号。为此，有以下按键可用。触碰按键将显示下一个。 • 自由文本: 显示在其中可输入 KRL 的栏。 • 用户列表: 显示带自定义变量的列表。前提条件：配置了该列表。 • 系统列表: 重新显示默认选择。

序号	说明
4	- 如果从默认选择中选择一个信号：输入编号。 - 如果从用户列表中选择了一个变量：输入值。
5	只在从默认选择中选择一个信号时，该栏才存在。如果信号已有注释则会显示出来。 从专家用户组起：可以输入名称或更改现有的名称。只要还没有保存行指令，则可以通过点击 长文本 重新复位这些更改。
6	CONT：指令被预进指针执行 [空]：指令触发预进停止
7	通过相应的按钮添加 ELSE 。 焦点必须在栏 1 或 2 上。

“更改”时的限制

通过按钮 **更改** 只能更改条件。ELSE 或 CONT 无法添加或删除。

14.4 SWITCH- CASE 编程

SWITCH- CASE 分支说明

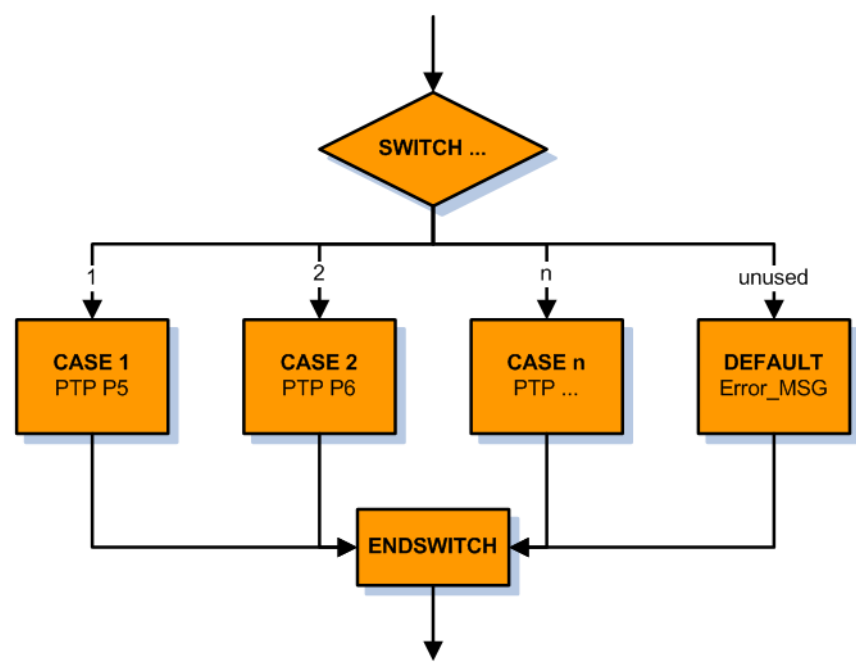


图 14-13：程序流程图： 分配模块 SWITCH - CASE

- 若需要区分多种情况并为每种情况执行不同的操作，则可用 SWITCH - CASE 指令达到目的。
- SWITCH- CASE 分支是一个分支或多重分支，并且用于不同情况。
- SWITCH 指令中传递的变量用作开关，在指令块中跳到预定义的 CASE 指令中。
- 如果 SWITCH 指令未找到预定义的 CASE，而 DEFAULT（默认）段事先已定义，则运行此段。

句法

```
SWITCH 选择标准
CASE 值
    指令
CASE 值
    指令
CASE 值
    指令
...
DEFAULT
    指令
ENDSWITCH
```

有关简单数据类型的值传递

- SWITCH- CASE 可与以下数据类型结合使用：
 - INT（整数）
示例含定义的 switch-case 分支，无替代路径

```
DEF MY_PROG( )
DECL INT my_number
...
INI
my_number = 2
...
SWITCH my_number
CASE 1
    SPTP P21 Vel=30% PDAT5 Tool[1] Base[1]
CASE 2
    SPTP P22 Vel=30% PDAT5 Tool[1] Base[1]
CASE 3
    SPTP P23 Vel=30% PDAT5 Tool[1] Base[1]
ENDSWITCH
...
```



当 *number* 不等于 1 或 2 或 3 时，则直接跳至 ENDSWITCH，不执行指令。

- CHAR（单个字符）
示例含定义的 switch-case 分支和一种替代路径

```
DEF MY_PROG( )
DECL CHAR my_sign
...
INI
my_sign = "a"
...
SWITCH my_sign
CASE "a"
    SPTP P21 Vel=30% PDAT5 Tool[1] Base[1]
CASE "b"
    SPTP P22 Vel=30% PDAT5 Tool[1] Base[1]
CASE "c"
    SPTP P23 Vel=30% PDAT5 Tool[1] Base[1]
DEFAULT
    SPTP P24 Vel=30% PDAT5 Tool[1] Base[1]
ENDSWITCH
...
```



当 *my_sign* 不等于 “a” 或 “b” 或 “c” 时，跳入 DEFAULT，以执行其指令。

一个分支中有多种解决方案

一个分支和一个替代路径中有多种解决方案的示例

```
SWITCH number
CASE 1,2
    SPTP P21 Vel=30% PDAT5 Tool[1] Base[1]
CASE 3,4,5
    SPTP P22 Vel=30% PDAT5 Tool[1] Base[1]
CASE 6
    SPTP Px3Vel=30% PDAT5 Tool[1] Base[1]
DEFAULT
    SPTP P24 Vel=30% PDAT5 Tool[1] Base[1]
ENDSWITCH
```



当编号不等于 3、4 或 5 时，则直接跳入第二个 “CASE”，以执行其指令。

用行指令对分支编程 (SWITCH- CASE)

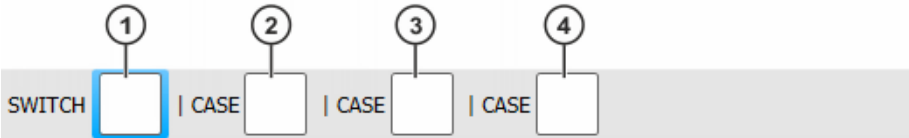


图 14-14: 行指令 SWITCH ... CASE

序号	说明
1	变量的选择标准 - 可能是 INT 或 CHAR 变量，但是不必事先声明。 - ENUM 变量必须事先或事后手动声明。
2	指令块的标记 标记的数据类型必须与变量（项号 1）的数据类型一致。 一个指令块可以有任意多的标记。CASE 栏中的多个标记必须通过逗号隔开。
3	如项号 2。
4	现有的 CASE 栏必须用标记填充或删除。 通过相应的按键可以删除或添加 CASE 栏。

“更改”时的限制

通过按键 **更改** 无法添加或删除 CASE。

可能的更改：

- 如果光标位于带 SWITCH 的行中，则可以更改变量。
- 如果光标位于带 CASE 的行中，则可以更改值。

14.5 编程跳转命令

说明

跳至程序中指定的位置。程序在该位置上继续运行。
跳转目标必须位于与 GOTO 指令相同的程序段或者函数中。

下列跳转是不可行的：

- 从外部跳至 IF 指令。
- 从外部跳至循环语句。
- 从一个 CASE 指令跳至另一个 CASE 指令。



在无条件的跳转时，程序不检查是否已经满足特定的条件。始终执行跳转。必须通过编程拦截可能出现的意外跳转。另见第二个编程示例。
用 GOTO 编程设计会使程序不清晰明了。
最好是使用分支或循环。

句法

```
...  
GOTO Marke  
...  
Marke:  
...
```

元素	说明
标记	跳转到的位置。在目标位置处在标签 结尾处必须有一个冒号。

示例

- 无条件跳转至程序位置 GLUE_STOP。

```
GOTO GLUE_STOP  
...  
GLUE_STOP:
```

- 通过用 IF 指令扩展，将无条件的跳转转换为有条件的跳转。跳至程序位置 GLUE_END。

```
IF X>100 THEN  
    GOTO GLUE_END  
ELSE  
    X=X+1  
ENDIF  
A=A*X  
...  
GLUE_END:  
END
```

14.6 用 KRL 语言给等待功能编程



图 14-15: WAIT, GO

用 KRL 为以下函数编程：

- 时间等待函数
- 信号等待函数

14.6.1 时间等待功能

KRL 中时间等待功能的说明



图 14-16：秒表

- 在过程可以继续运行前，时间等待功能等待指定的时间（time）。
- 句法

```
WAIT SEC time
```

用行指令对时间等待函数编程

可使用行指令便捷地编程设置等待时间。数值单位为秒。

```
WAIT Time= 2.5 [sec]
```

图 14-17：WAIT 行指令

插入的行指令本身是一个折叠夹，可以打开。相应的 KRL 指令显示为绿色。

```
7 → WAIT Time= 2.5 sec
8 WAIT SEC 2.5
```

图 14-18：打开的折叠

时间等待函数原理

- 时间等待函数的单位为秒（s）
- 最长时间为 2147484 秒，相当于 24 天多



时间等待函数的联机表单最多可等待 30 秒

- 时间值也可用一个合适的变量来确定
- 最短的有意义的时间单元是 0.012 秒（插补节拍）
- 如果给出的时间为负值，则不等待
- 时间等待函数触发预进停止，因此无法轨迹逼近
- 为了直接生成预进停止，可使用指令 WAIT SEC 0

给时间等待功能编程

- 具有固定时间的等待功能

```
SPTP P1 Vel=100% PDAT1
SPTP P2 Vel=100% PDAT2
WAIT SEC 5.25
SPTP P3 Vel=100% PDAT3
```

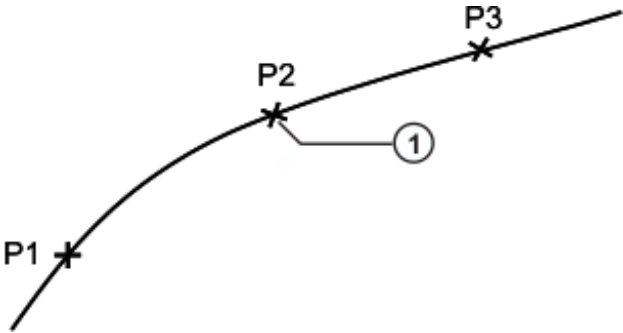


图 14-19：逻辑运动示例

序号	备注
1	在点 P2 处中断运动 5.25 秒钟

- 具有计算出时间的时间等待功能

```
WAIT SEC 3*0.25
```

- 具有变量的时间等待功能

```
DECL REAL time
time = 12.75
WAIT SEC time
```

14.6.2 信号等待功能

说明

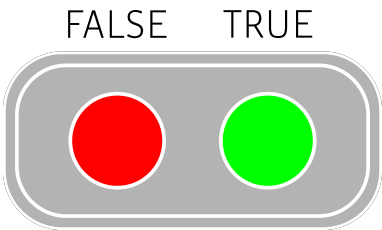


图 14-20：TRUE，FALSE

- 信号等待功能在满足条件（condition）时才切换到继续进程，使过程得以继续
- 句法

```
WAIT FOR condition
```

信号等待函数原理

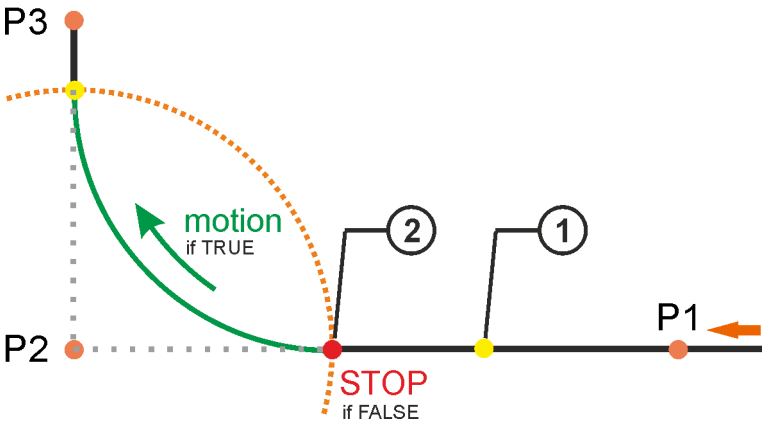


图 14-21：轨迹逼近运动时的 WAIT FOR

- 信号等待函数触发预进停止，因此无法轨迹逼近
- 尽管已满足了条件，仍生成预进停止
- 若在程序行中，指令 CONTINUE 被直接编程于等待指令之前，则当条件及时得到满足时就可以阻止预进停止



即使是一个空行，CONTINUE 也始终指的是下一行。

信号等待函数的编程

- 带预进停止的 WAIT FOR

```
SPTP P1 Vel=100% PDAT1
SPTP P2 CONT Vel=100% PDAT2
WAIT FOR $IN[20]==TRUE
SPTP P3 Vel=100% PDAT3
```

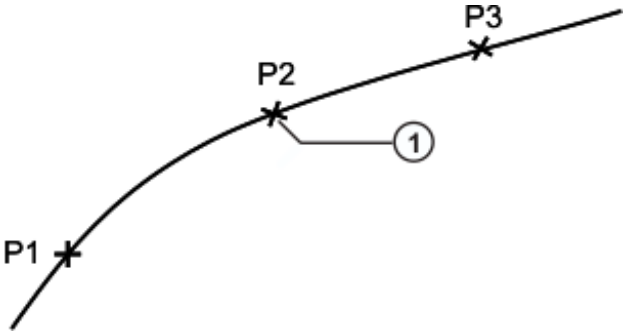


图 14-22：逻辑运动示例

	备注
1	运动在 P2 点中断。精确暂停后对输入端 20 进行检查。如果输入端状态正确，则可直接继续运行，否则会等待正确状态。

- 带预进中处理的 WAIT FOR （使用 CONTINUE）

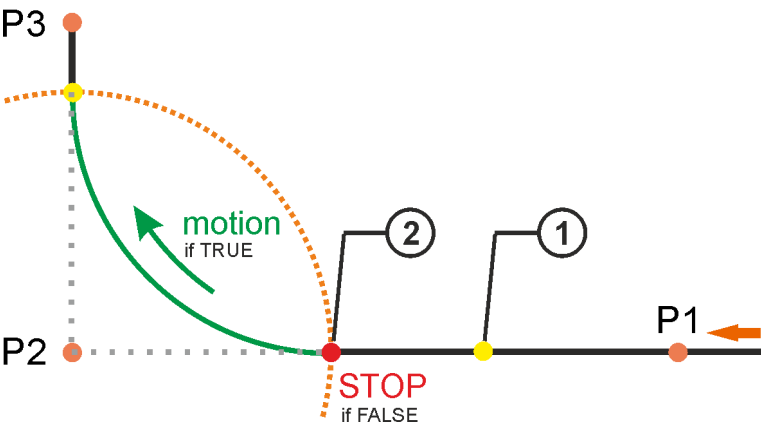


图 14-23：轨迹逼近运动时的 WAIT FOR

```
SPTP P1 Vel=100% PDAT1
SPTP P2 CONT Vel=100% PDAT2
CONTINUE
WAIT FOR ($IN[10] OR $IN[20])
SPTP P3 Vel=100% PDAT3
```

	动作
1	输入端 10 或者输入端 20 预进成或已预进成 TRUE。所以发生轨迹逼近
2	- 若前不久刚满足条件，则进行轨迹逼近运动 - 如果条件满足过迟，则无法进行轨迹逼近运动并在轨迹逼近极限处停止。机器人在轨迹逼近极限处等待，直到条件已满足。对此在信息提示窗口中显示“等待（输入端 10 或者输入端 20）。在测试运行方式（T1 或 T2）下，“模拟”按键可供使用。

用行指令对与信号有关的等待函数编程

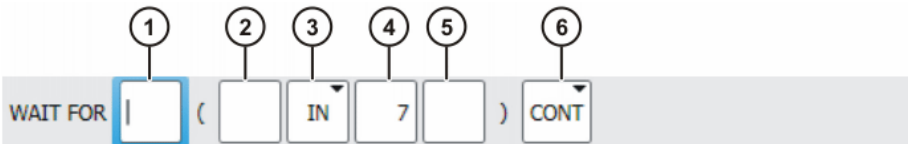


图 14-24：行指令 WAIT FOR

序号	说明
1	- NOT: 添加 NOT。 [空] 添加括号外部逻辑运算。该项位于两个括号的表达式之间。 通过相应的按键添加所需的项： - AND、OR 或 EXOR
2	- NOT: 添加 NOT。 [空] 添加括号内部逻辑运算。运算符位于一个括号的表达式内。 通过相应的按键添加所需的运算符： - AND、OR 或 EXOR

序号	说明
3	等待的信号。默认选择： • IN、OUT、CYCFLAG、TIMER 或 FLAG 替代默认选择，通过按键 用户列表 可以显示一个带自定义变量的列表。前提条件：配置了该列表。 系统列表 重新显示默认选择。通过触碰按键可以在它们之间切换。
4	• 如果从默认选择中选择一个信号：输入编号。 • 如果从用户列表选择一个变量：输入值。
5	只在从默认选择中选择一个信号时，该栏才存在。如果信号已有注释则会显示出来。 从专家用户组起：可以输入名称或更改现有的名称。只要还没有保存行指令，则可以通过点击长文本重新复位这些更改。
6	• CONT：指令被预进指针执行 • [空]：指令触发预进停止

14.7 练习：循环技术

练习内容

打开课程练习本并完成以下练习：

- 循环技术

15 使用上级控制系统作业

15.1 学习单元：使用上级控制系统作业

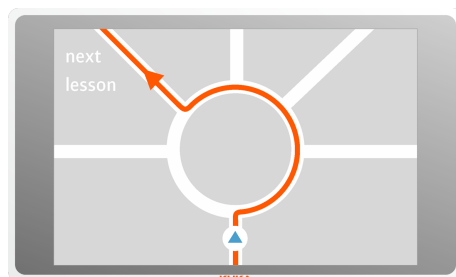


图 15-1：导航器

本学习单元讲授以下内容：

- 由 PLC 启动程序的准备工作
- 修改链接 PLC 外部自动端口程序 (Cell.src)

15.2 通过 PLC 启动程序的准备工作

机器人与系统相连

如果机器人进程应由一个中央位置进行控制（如由一个主导计算机或 PLC），则这一控制通过外部自动运行接口进行。



图 15-2：PLC 连接

PLC 和机器人之间进行通信需要用到已配置的现场总线，例如 PROFINET。通过该总线传输机器人进程的相关信号，作为数字输入和输出端。在机器人侧通过“外部自动运行”接口实现接口协议。

“外部自动运行”接口

- **发送至机器人的控制信号（输入端）：**
上级控制系统（PLC）向机器人控制器传输机器人进程的相关信号（如运行许可、故障确认、程序启动等）。
- **机器人状态（输出端）：**
机器人控制器向上级控制系统传输有关运行状态和故障状态的信息（如紧急停止、运行模式、起始位置、HomePosition 等）。

此外还需要：

- 已适配的 **CELL.SRC**
(从外部选择机器人程序的控制程序)
- 以及选择**外部自动运行模式**。
(在该运行模式下由一台主机或者 PLC 控制机器人系统)

从外部启动程序的安全须知

选择了 CELL 程序后必须以运行方式 T1 或 T2 执行 BCO 运行。



警告

- 如果选定的运动语句包括 PTP 运行指令，则 BCO 运行将作为 PTP 运动从实际位置移动到目标位置。
- 如果选定的运动语句包括 LIN 或 CIRC，则 BCO 运行将作为 LIN 运动被执行。
- 观察此运动，防止碰撞。在 BCO 运行中会自动降低速度。

如果已执行了 BCO 运行，则在外部的启动时便不再执行 BCO 运行。



警告

- 在外部的自动运行模式下没有 BCO 运行。
- 这表明，机器人在启动之后以编程的速度（没有减速）到达了第一个编程位置，并且在那里没有停止。

外部程序启动的操作步骤

前提条件

- 运行方式 T1 或 T2
 - 外部自动运行的输入/输出端及程序 CELL.SRC 已设置完毕。
1. 在导航器中选择 CELL.SRC 程序。
CELL 程序始终在目录 KRC:\R1 下



图 15-3: cell.src

2. 将程序倍率设定为 100 %。这是推荐设定值。
该数值可以变化地根据情况调整。

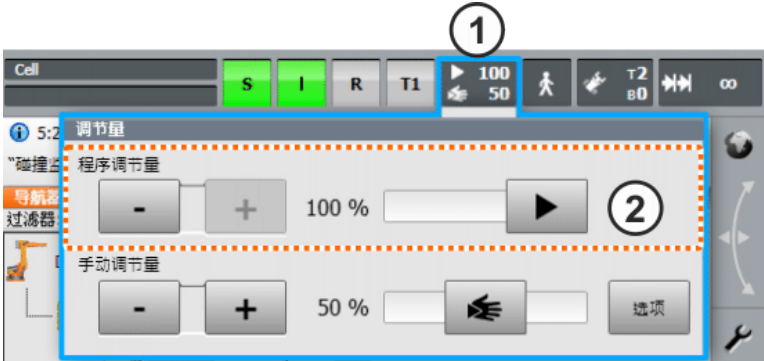


图 15-4: 程序倍率

3. 执行 BCO 运动：
按住确认开关。按住启动按键，直至信息窗显示 “SAK 到达”。
4. 选择“外部自动”运行方式。
5. 程序从上一级控制系统（PLC）开始启动。

15.3 修改链接 PLC 外部自动端口程序 (Cell.src)

Cell.src 控制程序

- 管理由 PLC 传输的程序号时，需要使用控制程序 Cell.src。
- 该程序始终位于文件夹“KRC:\R1”中。
- 与任何常见的程序一样，Cell 程序也可以进行单独调整，但程序的基本结构必须保持不变。

Cell 程序的结构和功能

```

1  ;EXT EXAMPLE1 ( )
2  ;EXT EXAMPLE2 ( )
3  ;EXT EXAMPLE3 ( )
4
5 → INIT
6  BASISTECH INI
7  CHECK HOME
8  PTP HOME Vel= 100 % DEFAULT]
9  AUTOEXT INI
10 LOOP
11   P00 (#EXT_PGNO,#PGNO_GET,DMY[],0 )
12   SWITCH PGNO ; Select with Programnumber
13
14   CASE 1
15     P00 (#EXT_PGNO,#PGNO_ACKN,DMY[],0 ) ; Reset
16     Progr.No.-Request
17     ;EXAMPLE1 ( ) ; Call User-Program
18
19   CASE 2
20     P00 (#EXT_PGNO,#PGNO_ACKN,DMY[],0 ) ; Reset
21     Progr.No.-Request
22     ;EXAMPLE2 ( ) ; Call User-Program
23
24   CASE 3
25     P00 (#EXT_PGNO,#PGNO_ACKN,DMY[],0 ) ; Reset
26     Progr.No.-Request
27     ;EXAMPLE3 ( ) ; Call User-Program
28
29   DEFAULT
30     P00 (#EXT_PGNO,#PGNO_FAULT,DMY[],0 )
31   ENDSWITCH
32 ENDLOOP

```

图 15-5: Cell 程序

1	初始化和 Home 位置 - 初始化基坐标参数 - 根据“Home”位置检查机器人位置 - 初始化外部自动运行接口 - 运行至“Home 位置”
2	无限循环： - 通过模块“P00”查询程序号 - 进入已经确定程序号的选择循环。
3	程序号的选择循环 - 根据程序号（保存在变量“PGNO”中）跳转至相应的分支（“CASE”）中。 - 记录在 CASE 中的机器人程序即被运行。 - 无效的程序号会导致程序跳转至“Default”分支中。 - 运行成功结束后会自动重复这一循环。

操作步骤

调用现有机器人程序

1. 切换到“专家”用户组。
2. 打开 CELL.SRC。
3. 在“CASE”段中将名称“EXAMPLE”用应从相应的程序编号调出的程序的名称替换。删除名称前的分号。

```
LOOP
  P00 (#EXT_PGNO,#PGNO_GET,DMY[],0 )
  SWITCH PGNO ; Select with Programnumber

  CASE 1
    P00 (#EXT_PGNO,#PGNO_ACKN,DMY[],0 )
    main()

  CASE 2
    P00 (#EXT_PGNO,#PGNO_ACKN,DMY[],0 )
    body_38()
    body_515()

  DEFAULT
    P00 (#EXT_PGNO,#PGNO_FAULT,DMY[],0 )
  ENDSWITCH
ENDLOOP
```

图 15-6：经适配的 Cell 程序举例

4. 关闭程序并保存更改。



必须与 PLC 程序员一起比对机器人程序及与之相关联的程序编号。

15.4 问题：上级控制系统上的工作

练习内容

打开课程练习本并回答以下问题：

- 在上级控制系统上作业

16 附录

16.1 自动运行外部接口的说明

说明

使用自动外部接口的输入/输出端

接口处重要信号概览

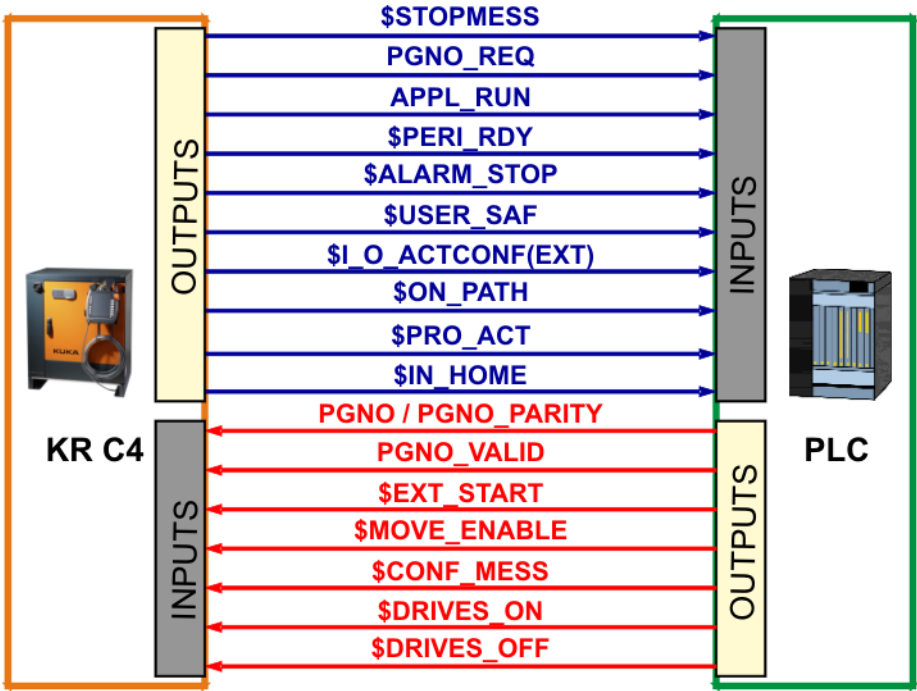


图 16-1：外部自动运行中最重要信号概览

输入端（从机器人控制器的角度）

- PGNO_TYPE - 程序号类型
此变量确定了以何种格式来读取上级控制系统传送的程序编号。

值	说明	示例
1	以二进制数值读取。 上级控制系统以二进制编码整数值的形式传递程序编号。	0 0 1 0 0 1 1 1 => PGNO = 39
2	以 BCD 值读取。 上级控制系统以二进制编码小数值的形式传递程序编号。	0 0 1 0 0 1 1 1 => PGNO = 27
3	以“N 选 1”的形式读取*。 上级控制系统或外围设备以“N 选 1”的编码值传递程序编号。	0 0 0 0 0 0 0 1 => PGNO = 1 0 0 0 0 1 0 0 0 => PGNO = 4

* 采用这种传递格式时，未对 PGNO_REQ、PGNO_PARITY 以及 PGNO_VALID 的值进行分析，因此无意义。

- **PGNO_LENGTH** - 程序号长度

此变量确定了上级控制系统传送的程序编号的位宽。数值范围：
1 ... 16。

若 PGNO_TYPE 的值为 2，则只允许位宽为 4、8、12 和 16。

- **PGNO_PARITY** - 程序号的奇偶位

上级控制系统传递奇偶位的输入端。

输入	功能
负值	奇校验
0	无校验
正值	偶校验

如果 PGNO_TYPE 值为 3，则 PGNO_PARITY 不被分析。

- **PGNO_VALID** - 程序号有效

上级控制系统传送读取程序号指令的输入端。

输入	功能
负值	在信号的脉冲下降沿应用编号。
0	在线路 EXT_START 处随着信号的脉冲上升沿应用编号。
正值	在信号的脉冲上升沿应用编号。

- **\$EXT_START** - 外部启动

设定了该输入端后，输入/输出接口激活时将启动或继续一个程序（一般为 CELL.SRC）。



仅分析信号的脉冲上升沿。



警告

在外部自动运行模式下**没有 BCO 运行**。这表明，机器人在启动之后以编程的速度（没有减速）到达了第一个编程位置，并且在那里没有停止。

- **\$MOVE_ENABLE** - 允许运行

该输入端用于由上级控制器对机器人驱动器进行检查。

信号	功能
TRUE	以运行方式 T1 及 T2 可手动运行。 可执行程序。
FALSE	停住所有驱动装置并锁定所有激活的指令



当驱动装置由上级控制器停住后，将显示“开通全部运行”的信息。删除了该信息并且重新发出外部启动信号后机器人才能重新运动。



投入运行时变量 \$MOVE_ENABLE 常常设计为值 \$IN[1025]。如果没有配置成另一个输入端，则不能外部启动。

- **\$CONF_MESS** - 确认信息

通过给该输入端赋值，当故障原因排除后，上级控制器将自己确认故障信息。



仅分析信号的脉冲上升沿。

- **\$DRIVES_ON** - 驱动装置接通
如果在此输入端上施加了持续至少 20 毫秒的高脉冲，则上级控制系统会接通机器人驱动装置。
- **\$DRIVES_OFF** - 驱动装置关闭
如果在此输入端上施加了持续至少 20 毫秒的低脉冲，则上级控制系统会关断机器人驱动装置。

输出端（从机器人控制器的角度）

- **\$ALARM_STOP** - 紧急停止
该输出端将在出现以下紧急停止情形时复位：
 - 按下了 KUKA 控制面板（KCP）上的紧急停止按钮。（内部紧急关断）
 - 外部紧急停止



出现紧急停止时可从输出端 **\$ALARM_STOP** 和 **\$ALARM_STOP_INTERN** 的状态看出是哪种类型的紧急停止：

- 两个输出端均为 FALSE：在 smartPAD 上已触发了紧急停机
- **\$ALARM_STOP** 为 FALSE，**\$ALARM_STOP_INTERN** 为 TRUE：外部紧急停止

- **\$ALARM_STOP_INTERN** - 内部紧急停止信号协议
如果该输出端为 TRUE，则没有内部紧急停止。发生内部紧急停止时，机器人控制器会将该输出端赋值为 FALSE。



与 **\$ALARM_STOP** 一样

- **\$USER_SAF** - 操作人员防护装置/防护门
该输出端在打开护栏询问开关（运行方式 AUT）或放开使能开关（运行方式 T1 或 T2）时复位。
- **\$PERI_RDY** - 驱动装置处于待机状态
通过设定此输出端机器人控制器通知上级控制系统机器人驱动装置已接通。
- **\$ROB_CAL**
当所有机器人轴均完成零点标定时，始终给该输出端赋值。一旦有一根机器人轴删除零点，该输出端将重置。
- **\$STOPMESS** - 停止信息
该输出端由机器人控制器来设定，以向上级控制器显示出现了一条要求停住机器人的信息。（例如：紧急停止按钮、运行开通或操作人员防护装置）
- **\$I_O_ACTCONF** - 外部自动运行激活
选择了外部自动运行这一运行方式并且输入端 **\$I_O_ACT** 为 TRUE（一般始终设为 **\$IN[1025]**）后，输出端为 TRUE。
- **\$PRO_ACT** - 程序激活/正在运行
当一个机器人的程序激活时，始终给该输出端赋值。在处理一个程序或中断时，过程为激活状态。程序结束时只有在所有脉冲输出端和触发器均处理完毕之后才视为未激活。
- **PGNO_REQ** - 程序号询问
在该输出端信号变化时，要求上级控制器传送一个程序号。
如果 **PGNO_TYPE** 值为 3，则 **PGNO_REQ** 不被分析。
- **APPL_RUN** - 应用程序在运行中
机器人控制器通过设置此输出端来通知上级控制系统机器人正在处理有关程序。

- **\$IN_HOME** - 机器人位于起始位置 (HOME)
该输出端通知上级控制器机器人正位于其起始位置 (HOME)。
- **\$ON_PATH** - 机器人位于轨迹上
只要机器人位于编程设定的轨迹上，此输出端即被赋值。在进行了 BCO 运行后输出端 ON_PATH 即被赋值。输出端保持激活，直到机器人离开了轨迹、程序复位或选择了语句。但信号 ON_PATH 无公差范围，机器人一离开轨迹，该信号便复位。
- **\$NEAR_POSRET** - 围绕 \$POS_RET 的公差范围的信号协议
通过给该输出端赋值，机器人控制器告知上级控制系统：机器人位于一个围绕在 \$POS_RET 中保存的位置的球体内。上级控制系统可用该信息决定是否允许重新启动该程序。
用户可以在文件 \$Custom.dat 中通过变量 \$NEARPATHTOL 定义球体的半径。
- **\$ROB_STOPPED** - 机器人停止的信号协议
如果机器人停止，则给该输出端赋值。即使对于 WAIT 指令，在等待期间也设定该输出端。该信号是 \$PRO_MOVE 的逆转。

外部自动运行通讯原理

整个过程概览

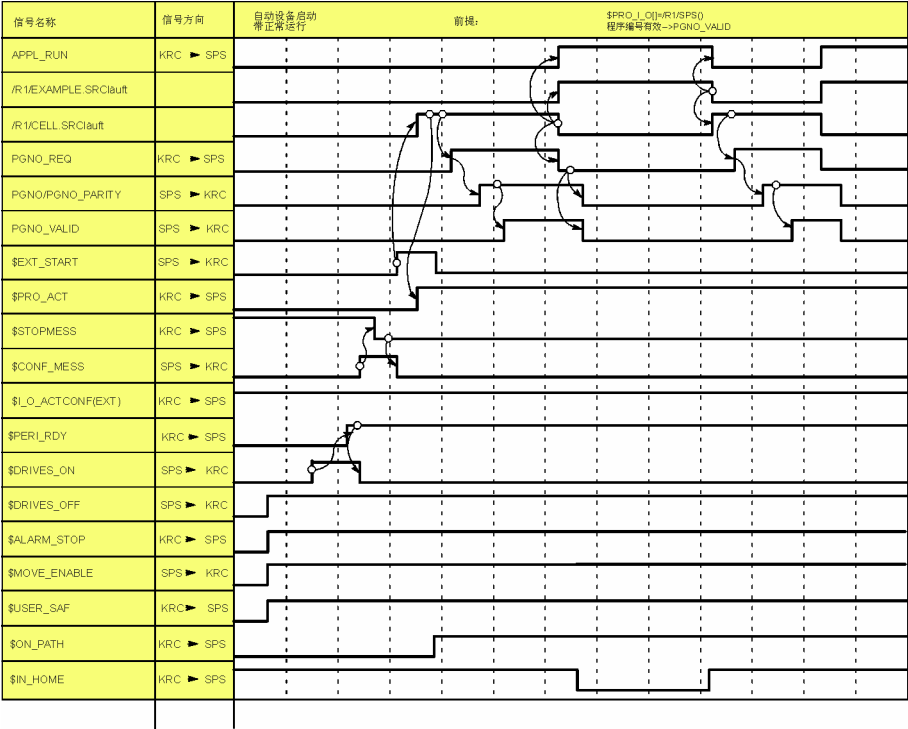


图 16-2：设备自动启动和正常运行，包括通过 PGNO_VALID 确认程序号

信号交换

分步任务划分

1. 接通驱动装置
(>>> “接通驱动装置” 页面 399)
2. 确认信息提示
(>>> “确认信息” 页面 400)
3. 启动 Cell 程序
(>>> “从外部启动程序 (CELL.SRC)” 页面 400)
4. 传递程序号并处理应用程序

(>>> “处理程序编码传输和应用程序” 页面 401)

每一项分步任务均设计为信号交换。必须满足相关条件，PLC 发送信号并且机器人控制系统将各机器人状态报告 PLC。



图 16-3: 信号交换

使用规定的信号交换是有益的。

接通驱动装置

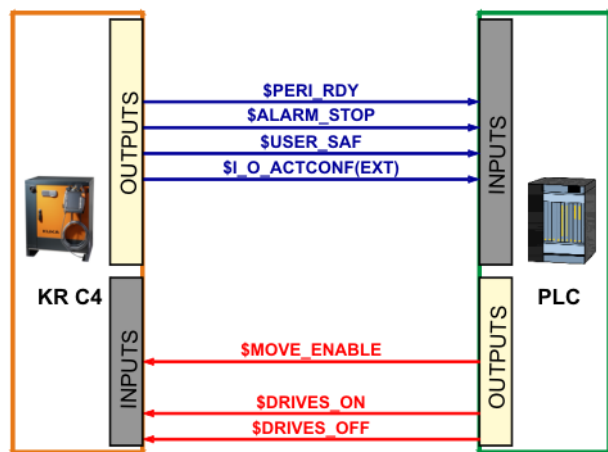


图 16-6

- 前提条件
 - \$USER_SAF - 防护门已关闭
 - \$ALARM_STOP - 无紧急停止
 - \$I_O_ACTCONF - 外部自动运行激活
 - \$MOVE_ENABLE - 允许运行
 - \$DRIVES_OFF - 未激活驱动装置关闭
- 接通驱动装置
 - \$DRIVES_ON - 接通驱动装置至少 20 ms 的信号
- 驱动装置处于待机状态
 - \$PERI_RDY - 一有驱动装置的反馈，信号 \$DRIVES_ON 便撤回

确认信息

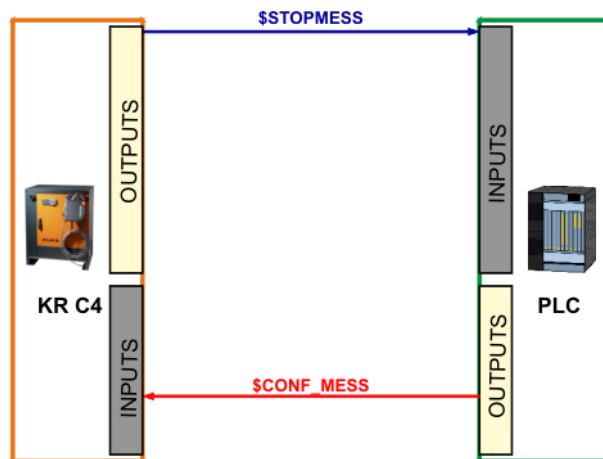


图 16-10

- 前提条件
 - \$STOPMESS - 有停止信息
- 确认信息提示
 - \$CONF_MESS - 确认信息提示
- 可确认的信息提示就此删除
 - \$STOPMESS - 不再有停止信息，现在可撤回 \$CONF_MESS

从外部启动程序 (CELL. SRC)

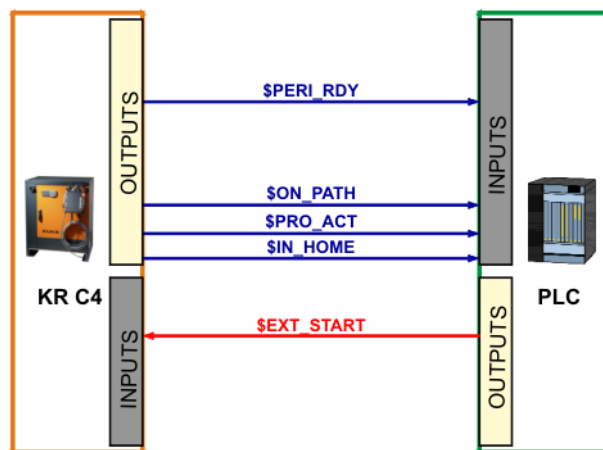


图 16-15

- 前提条件
 - \$PERI_RDY - 驱动装置处于待机状态
 - \$IN_HOME - 机器人位于起始位置 (HOME)
 - 无 \$STOPMESS - 无停止信息
- 外部启动
 - \$EXT_START - 接通外部启动 (脉冲正沿)
- CELL 程序在运行
 - \$PRO_ACT - 报告 CELL 程序在运行
 - \$ON_PATH - 一旦机器人位于编程设定的轨迹上，信号 \$EXT_START 便撤回

处理程序编码传输和应用程序

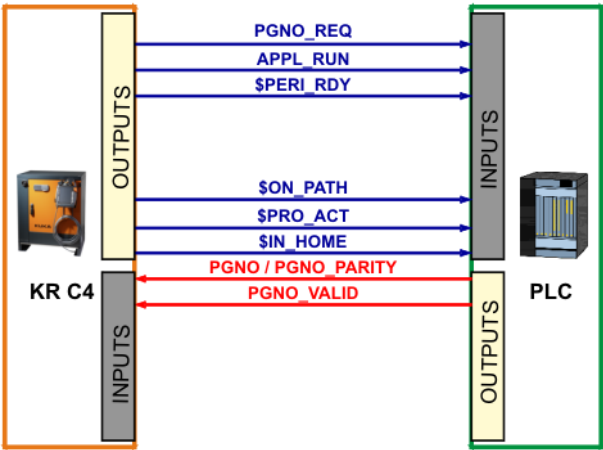


图 16-22

- 前提条件
 - \$PERI_RDY - 驱动装置处于待机状态
 - \$PRO_ACT - CELL 程序在运行
 - \$ON_PATH - 机器人在轨迹上
 - \$IN_HOME - 机器人位于起始位置 (HOME)，重新启动时不再需要
 - PGNO_REQ - 有程序号问询
- 程序号传递和确认
 - 程序号传递
 - (正确的数据类型 (PGNO_TYPE)，程序号长度 (PGNO_LENGTH) 和程序号的第一比特位 (PGNO_FBIT) 已设定)
 - PGNO_VALID - 将程序号切换为有效 (确认，脉冲正沿)
- 应用程序在运行中
 - APPL_RUN - 报告应用程序在运行中
 - 机器人离开起始位置。应用程序结束后机器人重又返回起始位置 (HOME)。

16. 1. 1 自动运行外部接口的配置

操作步骤

外部自动运行配置：输入端

1. 菜单路径：机器人按键 > 配置 > 输入/输出端 > 外部自动运行
2. 在数值栏中标定所需编辑的单元格，然后点击编辑 (C)。
3. 输入所需数值，并用 OK 保存。
4. 对所有待编辑的数值重复第 2 和第 3 步。
5. 关闭窗口。改动即被应用。



图 16-25：外部自动配置：输入端

序号	说明
1	编号
2	输入/输出端长文本名称
3	类型 - 绿色：输入/输出端 - 黄色：变量或系统变量（\$...）
4	信号或变量的名称
5	输入/输出端编号或信道编号
B	输出端根据内容分列在选项卡中。

外部自动运行配置：输出端

1. 菜单路径：KUKA 按键 > 配置 > 输入/输出端 > 外部自动运行
2. 通过按键输出端（B）(>>> 图 16-25) 切换到相应的概览。
3. 在数值栏中标定所需编辑的单元格，然后点击编辑。
4. 输入所需数值，并用 OK 保存。
5. 对所有待编辑的数值重复第 2 和第 3 步。
6. 按组排序所有数值。
 - 启动条件（A）



图 16-26：外部自动配置：输出端

• 程序状态 (B)



图 16-27：输出端，程序状态

• 机器人位置 (C)



外部自动配置：输出端

	名称	类	名称	值
1	位于起始位置	NO	\$IN_HOME	1000
2	第 1 个起始位置	NO	\$IN_HOME1	FALSE
3	第 2 个起始位置	NO	\$IN_HOME2	FALSE
4	第 3 个起始位置	NO	\$IN_HOME3	FALSE
5	第 4 个起始位置	NO	\$IN_HOME4	FALSE
6	第 5 个起始位置	NO	\$IN_HOME5	FALSE
7	机器人在轨迹上	NO	\$ON_PATH	997
8	机器人在轨迹附近	NO	\$NEAR_POSRET	147
9	机器人不在运行	NO	\$ROB_STOPPED	1023

启动条件 程序状态 机器人位置 运行方式

图 16-28：输出端，机器人位置

• 运行方式 (D)



外部自动配置：输出端

	名称	类	名称	值
1	测试 1 运行	NO	\$T1	993
2	测试 2 运行	NO	\$T2	994
3	自动运行	NO	\$AUT	995
4	外部自动运行	NO	\$EXT	996

启动条件 程序状态 机器人位置 运行方式

图 16-29：输出端，运行方式

依此打开各个选项卡并以已知的方式调整数值。

- 7. 关闭窗口。改动即被应用。

16.1.2 练习：配置外部自动运行接口（手动）

练习内容

打开课程练习本并完成以下练习：

- 配置外部自动运行接口（手动）

16.2 缩略语表

为了方便您使用控制系统，这里简要解释了部分最主要的缩写。

术语	说明
CCU	Cabinet Control Unit (控制柜控制单元)
CCU_SR	Cabinet Control Unit (控制柜控制单元) small robot (小型机器人)
CIB	Cabinet Interface Board (控制柜接口板)
CIB_SR	Cabinet Interface Board (控制柜接口板) small robot (小型机器人)
CBK	Customer Build Kinematic (客户专用运动系统)
CSP	Controler System Panel (控制系统操作面板)
Dual-NIC 双网卡	双工网卡
EDS	Electronic Data Storage (电子数据存储卡)
EGB	Elektrostatisch Gefährdete Bauteile (静电敏感部件)
EMD	Electronic Mastering Device (以前为 EMT, 是指用于机器人校准的电子控制装置)
EMC	Electro Magnetic Compatibility (电磁兼容性)
FSoE	Fail Safe over EtherCAT
GBE	Giga Bit Ethernet
编目	可包含不同的要素，例如模板、部件、运动系统。
KCB	KUKA Controller Bus (KUKA 控制器总线)
KEB	KUKA Extension Bus (KUKA 扩展总线)
KCP	KUKA Control Panel (可编程式手持操作器，现在叫 smartPAD)
KLI	KUKA Line Interface (KUKA 线路接口)
KOI	KUKA Operator Panel Interface (KUKA 操作面板接口)
KONI	KUKA Optional Network Interface (KUKA 可选网络接口)
KPC	KUKA 控制系统 PC
KPP	KUKA Power-Pack (KUKA 配电箱)
KPP_SR	KUKA Power-Pack (KUKA 配电箱) small robot (小型机器人)
KRF	Missing inline element 'emphasis'. Missing inline element 'emphasis'. Missing inline element 'emphasis'. 受控的机器人运行
KRL	KUKA Robot Language (KUKA 机器人语言)
KSB	KUKA System Bus (KUKA 系统总线)
KSP	KUKA Power-Pack (KUKA 伺服包)
KSP_SR	KUKA Power-Pack (KUKA 伺服包) small robot (小型机器人)

术语	说明
KSI	KUKA Service Interface (KUKA 服务接口)
HMI	Human Machine Interface (人机界面) KUKA.HMI 是 KUKA 操作界面。
LWL	Lichtwellenleiter (光缆)
MEMD	Micro Electronic Mastering Device (微电子控制装置)
MCC	Multi Core Computer (多核计算机)
MCFB	Motion Control Function Block (运动控制功能程序) 运动性作业任务编程的程序模块。该程序模块适合 PLC 并且是 KUKA 专用的。
MGU	Motor Gear Unit (电机齿轮箱单元) KUKA 运动系统电机齿轮箱组合单元
OPC	OLE (Object Link Embedding) for Process Control (用于过程控制的对象链接与嵌入)
OPI	Operator Panel Interface (操作面板接口, 即 smartPAD 的接口)
PDS	Power Drive System (动力驱动系统)
PMB	Power Management Board (电源管理板卡)
RCD	Residual Current Device (剩余电流保护断路器 (FI))
RDC	Resolver Digital Converter (分解器数字转换器)
SATA	Serial Advanced Technology Attachment (中央处理器与硬盘之间的数据总线)
SEMD	Standard Electronic Mastering Device (标准电子测量仪)
SIB	Safety Interface Board (安全接口板)
SBC	Single Brake Control (单制动控制)
STO	Safe Torque Off (安全关闭扭矩)
SION	Safety Input Output Node (安全输入/输出节点)
USB	Universal Serial Bus (用于连接电脑与附加设备的总线系统)
UPS	Unterbrechungsfreie Stromversorgung (不间断电源)
WorkVisual 编目编辑器	为 WorkVisual 创建编目条目的软件

索引

B

BCO..... 164

C

CBK..... 406
 CCU..... 406
 CCU_SR..... 406
 CIB..... 406
 CIB_SR..... 406
 CIRC 运动..... 198
 CSP..... 406

D

DECL..... 350, 353
 Dual-NIC 双网卡..... 406

E

EDS..... 406
 EMC..... 406
 EMD..... 406

F

FOR 循环..... 368
 FSoE..... 406

G

GBE..... 406
 GOTO..... 380
 GripperTech..... 295

H

HMI..... 407

I

IF..... 375

K

KCB..... 406
 KCP..... 406
 KEB..... 406
 KLI..... 406
 KOI..... 406
 KONI..... 406
 KPC..... 406
 KPP..... 406
 KRF..... 406

KRL..... 406
 KRL 中的数据管理..... 349
 KSB..... 406
 KSI..... 407
 KSP..... 406
 KUKA Xpert..... 35
 URL..... 36
 登录..... 38
 注册..... 37
 KUKA 机器人的机械系统..... 10

L

LIN 运动..... 198
 Loop 循环..... 365
 LWL..... 407

M

MCC..... 407
 MCFB..... 407
 MEMD..... 407
 MGU..... 407

O

OPC..... 407
 OPI..... 407

P

PDS..... 407
 PLC 程序启动..... 389
 PLC 连接
 Cell.src..... 392
 PMB..... 407

R

RCD..... 407
 RDW..... 407
 Robroot..... 58

S

SATA..... 407
 SBC..... 407
 SEMD..... 407
 SIB..... 407
 SION..... 407
 smartPAD..... 15
 smartPAD 基本设置..... 25
 SPTP 运动..... 187
 STO..... 407
 STOP 0..... 49
 STOP 1..... 49
 STOP 2..... 49

U

UPS.....	407
USB.....	407

W

WAIT.....	263
WHILE.....	370
WorkVisual 编目编辑器.....	407

安

安全停止 STOP 0.....	48
安全停止 STOP 1.....	48
安全停止 STOP 2.....	48
安全停止 0.....	48
安全停止 1.....	48
安全停止 2.....	48
安全停止, 外部.....	32
安全运行停止.....	48

比

比较运算.....	359
-----------	-----

编

编程, 外部 TCP.....	333
编程抓手.....	306
编程子程序调用	348
编辑.....	174
编目.....	406

变

变量.....	349, 353
变量, 单个显示.....	258

标

标记线, 用于零点标定.....	103
标准函数.....	360

操

操纵.....	359
操作.....	359, 360
操作及轴相关的手动运行.....	57
操作人员防护装置.....	32

测

测量.....	139, 316, 321
测量工具.....	114
测量外部工具和机器人引导的工件.....	326

插

插上和拔下 smartPAD.....	21
---------------------	----

程

程序: 复制.....	174
程序: 删除.....	175
程序: 重命名.....	176
程序操作.....	197
程序流程控制.....	365
程序模块.....	174
程序启动.....	165

初

初始化.....	164, 356
----------	----------

创

创建.....	185
创建程序	
编辑器中的视图.....	338
浏览器中的视图.....	337
模板.....	335
找到并排除程序中的错误.....	339

存

存档.....	176
---------	-----

单

单轴点动.....	50
-----------	----

等

等待功能, 时间相关.....	382
等待功能, 信号相关.....	383
等待信息.....	44

点

点动, 基坐标系.....	148
点动, 世界坐标系.....	61

对

对话信息.....	44
-----------	----

法

法兰坐标系.....	58
------------	----

防

防护门.....	32
----------	----

分

分支 (SWITCH- CASE)..... 378

附

附录..... 395

更

更改, 运动命令..... 225

更改变量..... 258

工

工具负载数据..... 110

工具和基座管理..... 105

工具手动运行..... 138

工具坐标系..... 58

工作台的基坐标测量..... 147

固

固定工具..... 316

固定工具, 手动移动..... 327

关

关键词..... 351

轨

轨迹逼近: SCIRC..... 211

轨迹逼近: SLIN..... 211

轨迹逼近: SPTP..... 190

行

行指令..... 187

恢

恢复..... 176

机

机器人安全..... 31

机器人程序..... 161, 163

机器人单轴移动..... 50

机器人基础知识..... 9

机器人控制器 (V)KR C4..... 12

机器人控制系统的运行方式..... 46

机器人上的负载..... 109

机器人上的附加负载..... 111

机器人位置

smartPAD..... 76

显示..... 73

机器人系统文档..... 35

基

基本运算类型..... 359

基坐标..... 139

基坐标系..... 58

基坐标系中的手动运行..... 152

尖

尖触头工具测量..... 129

检

检查抓爪

行指令..... 309

简

简单的数据类型..... 364

简单数据类型..... 352

借

借助 6D 鼠标在世界坐标系中操作以及手动运

行..... 69

借助移动键在世界坐标系中操作以及手动运行 68

紧

紧急停止按钮..... 31

紧急位置..... 51

空

空运转程序..... 197

控

控制器信息读取查看..... 43

控制系统关机..... 157

练

练习. 57, 68, 69, 101, 129, 133, 138, 147, 152, 197, 219, 272, 312, 313, 326, 364, 387

使用 KUKA Xpert Basic..... 41

练习, 恒速运动区域和条件停止..... 294

练习, 样条..... 293

练习, 样条组..... 255

练习, 用固定工具手动移动..... 332

练习, 用外部 TCP 进行运动编程..... 334

练习, 执行机器人程序..... 169

练习: 配置外部自动运行接口 (手动)..... 405

零

零点标定..... 84

零点标定，带偏移量的负载零点标定..... 101

逻

逻辑..... 272
逻辑，概述..... 257
逻辑编程
简单的逻辑..... 262
逻辑运算..... 360

切

切换运行模式..... 49

确

确认信息..... 44

日

日志..... 180

上

上级控制系统上的工作..... 394

声

声明..... 354

使

使用
KUKA Xpert..... 39

世

世界坐标系..... 58

手

手动移动，固定工具..... 327
手动运行，工具坐标系..... 134

缩

缩进..... 344
缩略语表..... 406

提

提示信息..... 44

条

条件停止..... 280

跳

跳转..... 380
跳转命令..... 380

停

停机类别 0..... 49
停机类别 1..... 49
停机类别 2..... 49
停止反应..... 47

外

外部自动运行..... 389

腕

腕点..... 199, 200

位

位运算..... 360

问

问题 394

协

协定..... 349

信

信号输出功能，单一..... 267

学

学习单元
KUKA 机器人系统的结构和功能..... 9
变量和协议..... 349
程序运行控制的用处..... 365
创建和更改编程的运动..... 185
机器人投入运行..... 79
使用程序文件..... 171
使用工艺程序包..... 295
使用上级控制系统作业..... 389
外部工具的配置和编程..... 315
样条运动编程..... 233
移动机器人..... 43
在机器人程序中使用逻辑函数..... 257
执行机器人程序..... 161
专家界面介绍..... 335

循

循环..... 365
循环，REPEAT... UNTIL..... 373

循环, WHILE.....	370	抓爪编程“标牌”.....	312
循环技术.....	387	抓爪工具测量.....	133
沿		专	
沿轨迹运行和轨迹逼近.....	219	专家界面.....	335
样		状	
样条, 运动方式.....	233	状态信息.....	44
样条运动.....	233		
样条组.....	233	自	
 		自动外部.....	395
用		自动运行外部接口的配置.....	401
用户组.....	79	自动运行外部接口的说明.....	395
更改.....	82	自由旋转装置.....	54
切换.....	80		
 		坐	
优		坐标系.....	58
优先级.....	362		
由			
由机器人引导的工件.....	321		
语			
语句.....	375		
语句选择.....	253		
预			
预进.....	351		
原			
原理.....	84		
运			
运动指令.....	185		
增			
增量.....	54		
增量式手动运行.....	53		
折			
折叠.....	344		
抓			
抓手信号.....	297		
抓爪编程“笔”.....	313		