《工业机器人离线编程与仿真（ABB）》教材

展示目录

一、教材简介 2

1.1 教材基本信息 2

1.2 内容简介 3

二、教材配套资源预览 4

2.1 在线课程展示 4

2.2 在线课程资源与学习情况统计 4

2.3 部分资源展示 5

三、教材目录 7

四、样章试读（项目三） 11

一、教材简介

1.1 教材基本信息

教材名称	工业机器人离线编程与仿真（ABB）
主编	谢子明、李浩、谢楚雄
ISBN	9787548752318
出版社	中南大学出版社
出版时间	2023.2
定价	56.00 元

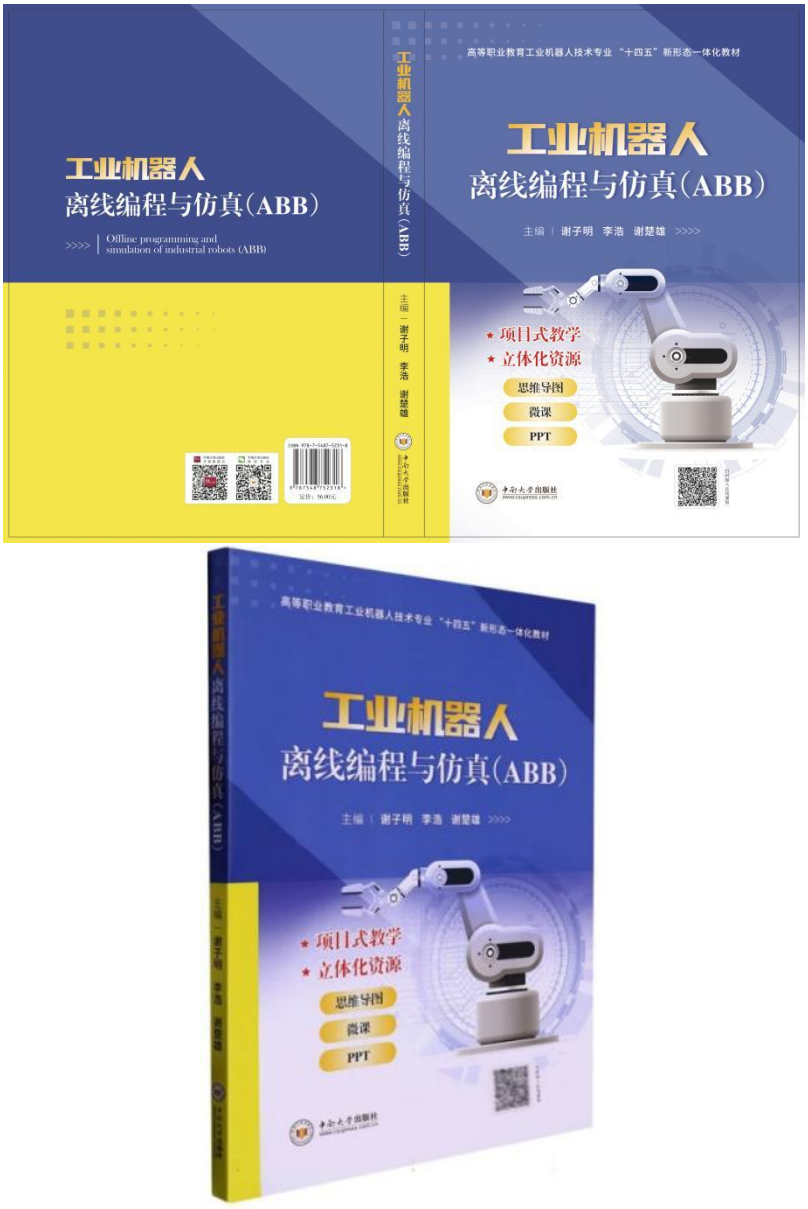


图 1-1 教材实物图

1.2 内容简介

本教材基于 ABB 工业机器人离线编程软件-RobotStudio,旨在通过搬运、码垛等多个工业机器人典型应用场景的虚拟仿真，培养工业机器人技术领域人才所需的离线编程与仿真相关的专业知识与技能。本教材为新形态融媒体、项目式教材，项目结合 1+X 考证与专业技能考核，并融入课程思政。各项目根据实施过程又细分多个教学任务，各任务依次递进，并将工业机器人常用运动指令、逻辑控制指令等理论知识融入各项目任务中，充分体现理实一体与任务驱动特点。同时，还开设有在线开放课程，包含学习指南、PPT、微课视频、项目打包文件、题库等丰富的教学资源，学生可以随时随地自主学习，教师也可以利用这些资源灵活组织课堂教学。此外，各项目还设立了拓展任务，方便教师拓展教学与学生拓展学习。

本教材共包含五个项目：项目一主要介绍工业机器人离线编程技术、主流专用型与通用型离线编程软件；项目二主要介绍基本运动指令的应用、坐标系创建、手动与自动路径创建、仿真辅助工具的应用等；项目三主要介绍建模工具、Smart 组件、I/O 指令、逻辑控制指令的应用等；项目四主要介绍动态传送带创建、数组、偏转函数的应用等；项目五主要介绍喷涂工具的创建与应用等。

二、教材配套资源预览

2.1 在线课程展示

课程二维码

图 2-1 在线精品课程截图

2.2 在线课程资源与学习情况统计

图 2-2 资源类型与数量统计

课程资源与学习数据(选择两(学)期)

数据项		第1学期	第2学期	第3学期	第4学期	第5学期	第6学期	第7学期
当前选课人数	选课人数(人)	634	1847	982	1329	244	1127	422
课程资源	数量(个)	76	379	259	259	259	259	259
视频资源	总数量(个)	32	98	62	62	62	62	62
	总时长(分钟)	430.40	555.50	464.00	464.00	464.00	464.00	464.00
3D图/虚拟仿真类资源	数量(个)	0	20	24	23	0	0	0
课程公告	数量(次)	1	8	1	5	2	4	2
检测和作业	总次数(次)	16	16	16	19	16	16	16
	习题总数(道)	140	140	116	184	184	183	131
	参与人数(人)	190	429	69	246	137	456	52
互动交流情况	发帖总数(帖)	2	18317	2	13	9	523	361
	教师发帖总数(帖)	2	17	2	9	18	69	9
	参与互动人数(人)	0	54	2	8	1	62	30
考核(试)	次数(次)	1	1	1	1	1	2	1
	试题总数(题)	31	31	41	41	41	145	41
	参与人数(人)	220	559	343	451	69	437	174
	考试通过人数(人)	189	460	293	365	57	337	137

图 2-3 各期开课学习情况统计

2.3 部分资源展示

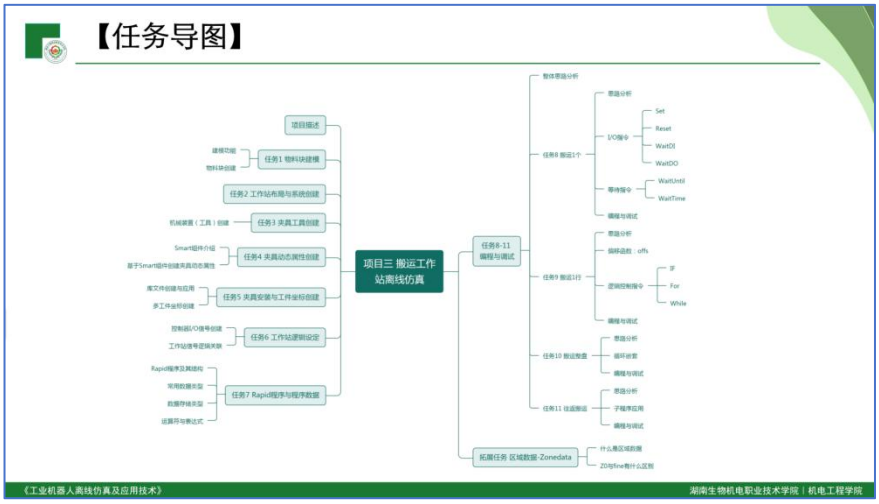


图 2-4　PPT 课件截图

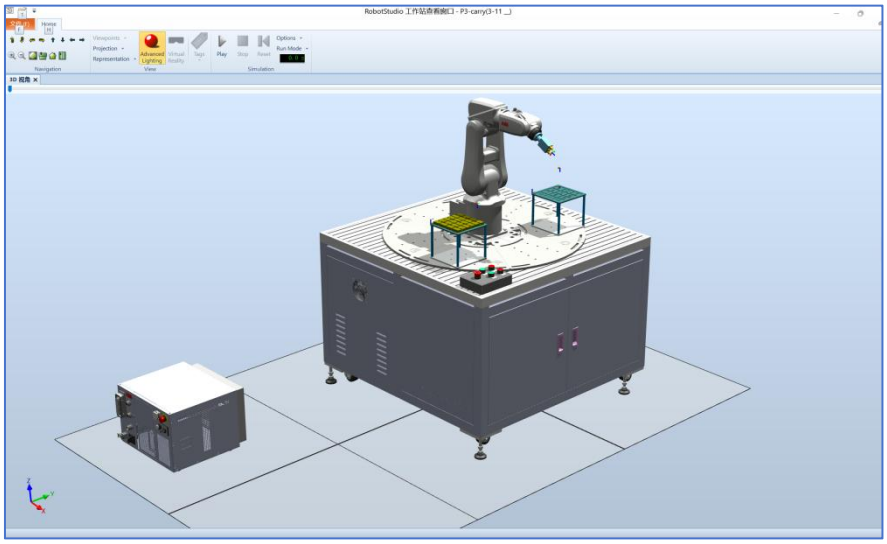


图 2-5 项目仿真截图

扫码观看夹具动态属性创建的操作视频

图 2-6 微课视频截图

微课视频展示地址：https://qr.csupress.com.cn/Public/ResourceList/Detail/43614

图 2-7 主题讨论截图

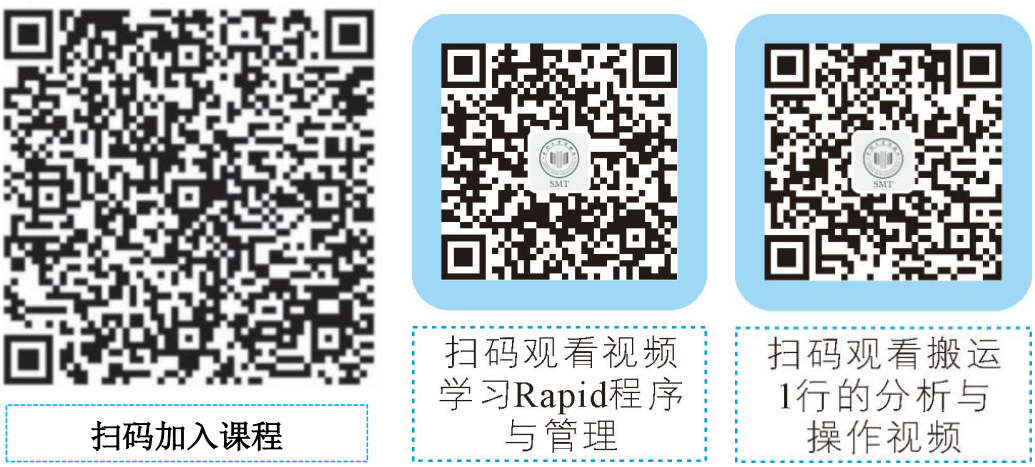


图 2-8 在线课程二维码与部分微课视频二维码

三、教材目录

目 录

项目一　工业机器人离线编程技术概述

任务 1　工业机器人离线编程技术 …… 3
1.1.1　工业机器人编程方式 …… 3
1.1.2　工业机器人离线编程技术的作用与特点 …… 5
1.1.3　工业机器人离线编程技术的应用 …… 6
任务 2　常用离线编程软件 …… 8
1.2.1　主流工业机器人品牌及其专用离线编程软件 …… 8
1.2.2　通用型离线编程软件 …… 12
1.2.3　常用离线编程软件对比 …… 16
任务 3　RobotStudio 软件下载与安装 …… 17
1.3.1　软件下载 …… 17
1.3.2　软件安装 …… 18
任务 4　软件界面与基本操作 …… 21
1.4.1　项目解包与打包 …… 22
1.4.2　软件界面 …… 24
1.4.3　键盘与鼠标基本操作 …… 25

项目二　工业机器人绘图工作站离线编程

任务 1　工业机器人绘图工作站搭建 …… 28
2.1.1　工作台模型导入及安装 …… 28
2.1.2　机器人选型及安装 …… 30

2.1.3 画笔工具导入及安装 …… 33
2.1.4 工件台导入及安装 …… 35
2.1.5 工业机器人系统创建 …… 37
任务 2 工业机器人简单绘图轨迹创建 …… 39
2.2.1 工业机器人手动操作 …… 45
2.2.2 工件坐标系创建 …… 58
2.2.3 创建绘图路径 …… 59
2.2.4 绘图路径验证 …… 64
2.2.5 仿真调试 …… 65
任务 3 工业机器人绘图轨迹自动创建 …… 66
2.3.1 路径自动创建 …… 66
2.3.2 目标点调整 …… 69
2.3.3 轴参数配置调整 …… 72
2.3.4 路径完善 …… 73
2.3.5 碰撞功能设置 …… 75
2.3.6 仿真调试 …… 77

项目三 工业机器人搬运工作站离线编程

任务 1 搬运工作站布局 …… 84
3.1.1 工作站创建 …… 84
3.1.2 物料块建模 …… 85
3.1.3 工作站布局 …… 88
3.1.4 系统创建 …… 98
任务 2 夹具工具创建 …… 100
3.2.1 夹爪安装 …… 100
3.2.2 机械装置(工具)创建 …… 101
任务 3 夹具动态属性创建 …… 104
3.3.1 Smart 组件介绍 …… 104
3.3.2 利用 Smart 组件创建夹具动态属性 …… 104
任务 4 工作站逻辑设定 …… 111
3.4.1 控制器 I/O 信号创建 …… 112
3.4.2 工作站信号逻辑关联 …… 113
任务 5 搬运 1 个 …… 114
3.5.1 思路分析 …… 114

3.5.2 I/O 指令 …… 115
3.5.3 程序等待指令 …… 116
3.5.4 工件坐标与点位创建 …… 117
3.5.5 路径程序创建 …… 121
3.5.6 同步 Rapid 与仿真运行 …… 124
任务 6 搬运 1 行 …… 125
3.6.1 思路分析 …… 125
3.6.2 Offs——偏移功能函数 …… 126
3.6.3 逻辑控制指令 …… 126
3.6.4 Rapid 程序与管理 …… 129
3.6.5 编程与调试 …… 134
任务 7 搬运整盘 …… 135
3.7.1 思路分析 …… 135
3.7.2 循环嵌套 …… 135
3.7.3 编程与调试 …… 136
任务 8 往返搬运 …… 137
3.8.1 思路分析 …… 137
3.8.2 子程序应用 …… 137
3.8.3 编程与调试 …… 137

项目四 工业机器人码垛工作站离线编程

任务 1 码垛工作站布局 …… 145
4.1.1 创建空工作站 …… 145
4.1.2 安装基座 …… 146
4.1.3 安装机器人 …… 150
4.1.4 安装吸盘工具 …… 151
4.1.5 安装安全围栏 …… 154
4.1.6 安装控制柜、空调、示教器等设备 …… 155
4.1.7 安装输送链 …… 157
4.1.8 安装托盘和工件 …… 158
任务 2 动态吸盘工具创建 …… 159
4.2.1 创建机器人系统 …… 159
4.2.2 创建动态吸盘工具 …… 161
任务 3 动态流水线创建 …… 174

4.3.1 动态传输链创建 …… 174
4.3.2 源对象创建 …… 175
4.3.3 队列对象创建 …… 176
4.3.4 线性运动对象创建 …… 177
4.3.5 平面限位传感器对象创建 …… 177
4.3.6 逻辑门对象创建 …… 178
4.3.7 属性与连结关系创建 …… 179
4.3.8 信号创建 …… 180
4.3.9 信号连接 …… 181
任务4 工件坐标与手动路径创建 …… 183
4.4.1 工件坐标创建 …… 183
4.4.2 创建简单抓取路径 …… 185
4.4.3 创建简单放置工件路径 …… 188
任务5 工作站逻辑设定 …… 190
4.5.1 数据总线下板卡的创建 …… 191
4.5.2 系统信号创建 …… 192
4.5.3 工作站逻辑信号与连接 …… 194
任务6 编程与调试 …… 196
4.6.1 添加逻辑指令 …… 196
4.6.2 偏移、偏转函数及数组的应用 …… 199
4.6.3 调试与仿真 …… 204

项目五 工业机器人喷涂工作站离线编程

任务1 喷涂工作站布局 …… 208
5.1.1 喷涂机器人的选择与创建 …… 209
5.1.2 喷涂工具的选择与创建 …… 210
5.1.3 传送带创建 …… 210
5.1.4 工件的创建 …… 211
5.1.5 工作站的布局 …… 212
任务2 喷涂组件的设置 …… 212
任务3 工作路径创建 …… 213
任务4 工作站逻辑设定 …… 215
任务5 仿真与调试 …… 220

四、样章试读（项目三）

项目三　工业机器人搬运工作站离线编程

项目描述

某企业要求使用工业机器人对该企业生产的产品进行搬运转移，将放置于托盘上的 12 个产品搬运至另一托盘。本项目基于 RobotStudio 软件构建如图 3-1 所示的搬运场景，并编程模拟该搬运作业。要求将工业机器人右侧物料盘上的物料块搬运至左侧物料盘，按 1→12 的顺序依次搬运，且保持物料位一一对应。

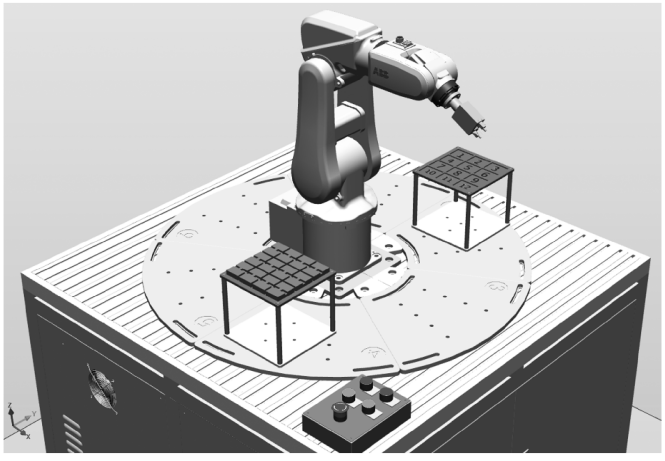


图 3-1　项目效果图

学习目标

◆ 知识目标

1. 掌握 RobotStudio 软件中建模工具的使用。
2. 了解 Smart 组件及其作用，熟练掌握其常用子组件的用法。
3. 熟练掌握 I/O 指令、程序等待指令、逻辑控制指令的功能与用法。
4. 熟练掌握偏移功能函数(offs)的功能与用法。
5. 了解 Rapid 程序的组成，掌握 Rapid 程序中常用程序数据、基本运算符的特点与用法。
6. 了解循环嵌套与子程序的应用方法。

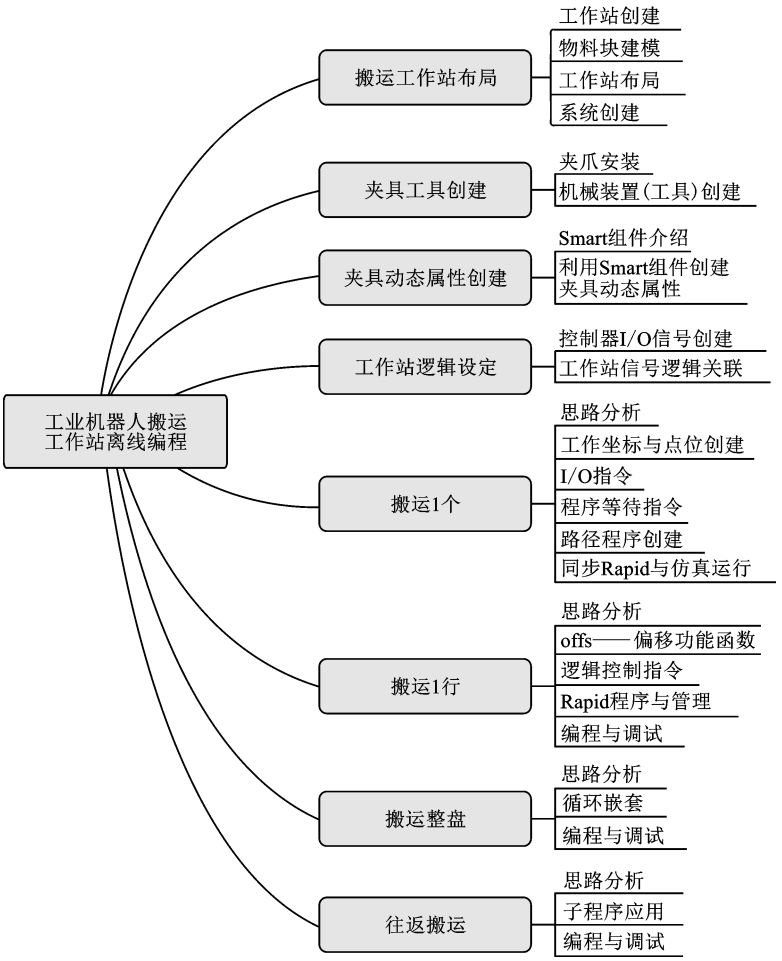
能力目标

1. 能利用建模工具创建一般复杂程度的三维模型。
2. 能熟练完成一般复杂程度的工作站布局。
3. 能利用 Smart 组件创建具备动态属性的工具或其他设备。
4. 能正确设置工作站中各信号的工作逻辑。
5. 能熟练应用 I/O 指令、程序等待指令、逻辑控制指令、偏移功能函数编写程序。
6. 能利用子程序进行结构化编程。

素质目标

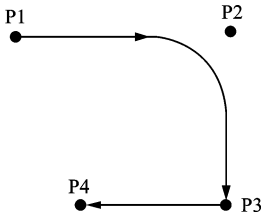
1. 具有良好的学习习惯、软件使用习惯。
2. 具有吃苦耐劳、严谨细致的工作作风。
3. 具备良好的三维空间意识。
4. 具备勤于思考、精益求精的工匠精神。
5. 具备良好的职业生涯规划意识。

知识图谱



课程思政

确保工作质量，提升工作效率



工业机器人的运动路径通常包括多个目标点位，通过运动指令控制机器人在各点位间执行直线、圆弧等运动路径。在执行运动路径时，机器人 TCP 会根据运动路径中的区域数据-ZoneDate（也称为转弯半径）确定是否精确到达某个点位，如果区域数据为“fine”，则 TCP 精确到达该点后再前往下一个点位，如果区域数据为“Z”，则程序会提前读取下一目标位置，当 TCP 在距离该点位一定距离时开始转弯，平滑过渡直接前往下一个目标点，如图中 P2 点。当区域数据为“fine”时，机器人 TCP 完全到达该位置后会短暂停止运转，然后再前往下一个目标点，如图中 P3 点。当区域数据为“Z”时，机器人 TCP 会根据设置的转弯半径直接平滑过渡至下一点，不仅会缩短实际的运行路径长度，而且不会停顿，因而会缩短路径执行时间，提高工作效率。

所以在进行实际应用时，要根据加工需要合理设置区域数据，如进行搬运作业时，拾取和放置物料的位置通常需要精确到达，应当使用“fine”，否则可能会使取放料操作失败或者位置出现偏差，从而影响作业质量。为了确保安全，前往取放料位置过程中通常需要在合适位置设置过渡点，这些过渡点一般不需要精确到达，可以使用“Z”，使路径更平滑且高效。这正如学习与工作，在确保学习与工作质量的前提下，我们要寻求合适的方法提高学习与工作效率。

“理无专在，而学无止境也”，学习是我们增长知识、提升技能、寻求真理的最佳途径。作为青年大学生，我们要把学习作为第一任务，并且掌握正确的学习方法，保持高效的学习效率。那如何提高自己的学习效率呢？不妨从以下几个方面进行尝试。

1. 制定小目标

学习开始时不能想着制定一个最终目标，然后就不管不顾了。在制定了最终目标后应该把自己的目标分成一个个小目标来逐一实现，比如，日目标、周目标、月目标、年目标等。根据制定的小目标列出目标清单，记录好自己实现目标的进度，进而再制定每一个小目标的学习计划，这样每完成一个小目标就会激励自己完成下一个目标。通过努力就会距离最终目标越来越近，而且还可以慢慢培养自己良好的学习习惯和学习状态。

2. 管理好学习时间

一天中 24 小时对于每一个人意义都是不一样的，有些人早上学习状态很好，而有些人则是下午或者晚上。我们需要合理分配学习与休息时间，尽可能多地把时间投入到学习中来，同时选择自己精神状态良好的时间段来学习。

3. 保持良好的生活习惯

想要提高学习效率，就要确保睡眠质量与充足的睡眠时间。因此，我们需要养成规律的作息习惯，做到早睡早起，不随意熬夜。同时，还需要保持健康的饮食习惯，确保充足的营养供应，以保证良好的身体条件。

4. 选择合适的辅助工具与学习方法

工欲善其事，必先利其器，选择适合自己的工具能使得事半功倍。同样，好的学习方法能为我们节约不必要的时间。殊途同归，我们当然要选择最便捷的路。

5. 善于思考和总结

“学而不思则罔”，我们要时常进行学后思考与总结，这样才能发现知识之间的异同与联系，思考和总结其实也是对新学知识的消化过程。总结可以安排在每天晚上睡觉前进行，这样和第二天的学习有良好的衔接。

项目实施

任务1　搬运工作站布局

3.1.1　工作站创建

启动软件后，依次选择“文件→新建→空工作站解决方案”，设置好解决方案名称与位置路径，最后点击创建，即可在设定的路径下自动创建该工作站项目文件并保存，如图 3-2 所示。

图 3-2　创建工作站

3.1.2　物料块建模

扫码观看物料块建模操作视频

工业机器人离线编程中复杂、精致的三维模型通常通过第三方专业三维建模软件创建好后导入，但是，Robotstudio 软件也具备简单三维建模的功能，可以创建固体(包括矩形体、圆锥体、圆柱体、锥体、球体等)、表面(包括表面圆、表面矩形、表面多边形、从曲线生成表面等)、曲线(包括直线、圆、弧线、椭圆、矩形、多边形等)，也可以将多个模型进行交叉、减去、结合等生成新的模型。接下来以创建如图 3-3 所示的物料块为例介绍 RobotStudio 三维建模功能的使用。该物料为在 55 mm×40 mm×15 mm 的大矩形体上切除两个 16 mm×8 mm×5 mm 小矩形体得到，因此，可以先创建一大一小两个基本模型，然后再对两个基本模型进行减去组合得到所需物料块模型，详细步骤如表 3-1 所示。

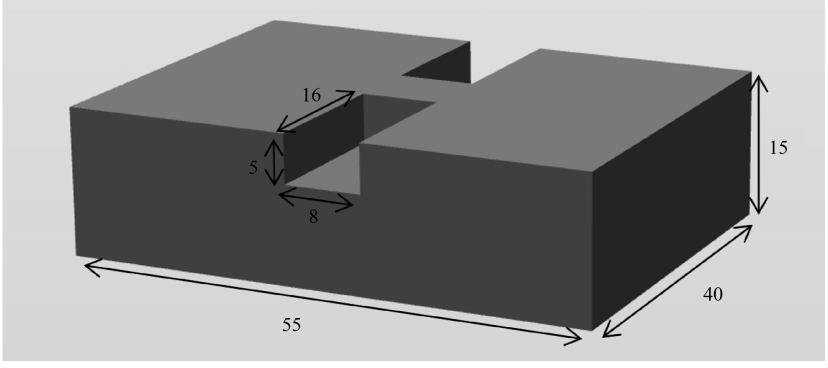
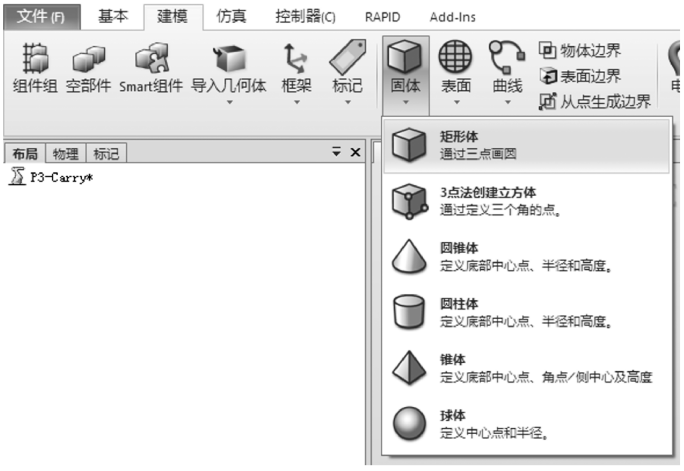
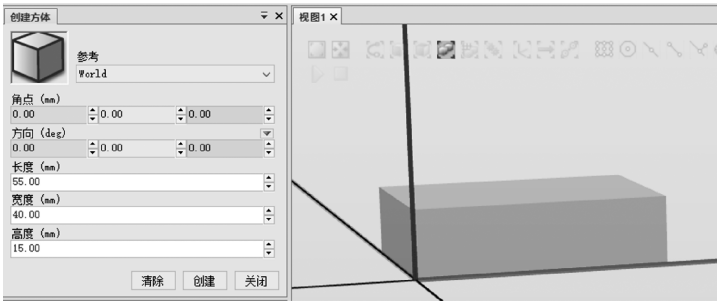
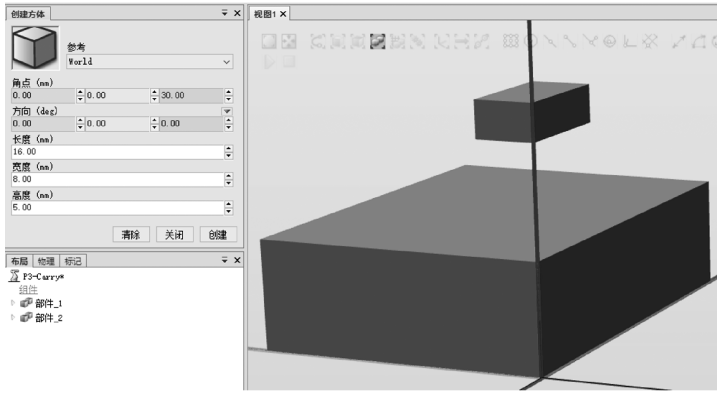
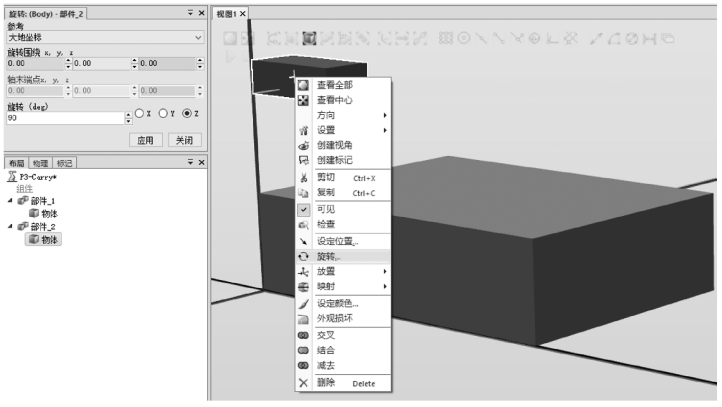


图 3-3　物料块模型(单位：mm)

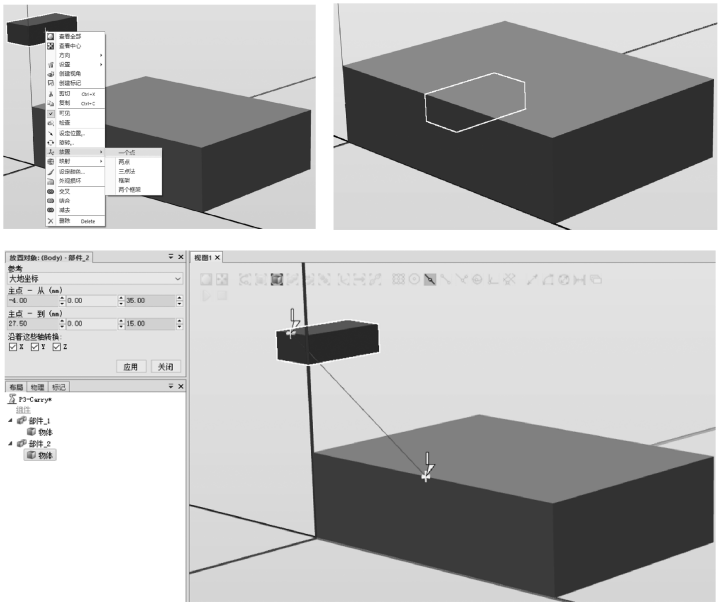
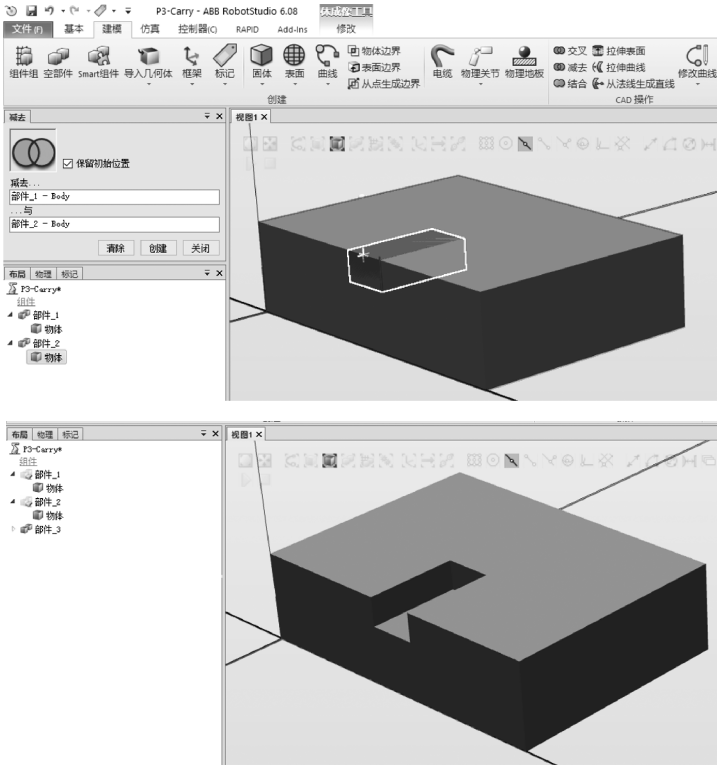
表 3-1　创建物料块操作步骤

图片示例	步骤说明
	步骤 1：创建大矩形体 说明：依次选择“建模→固体→矩形体”

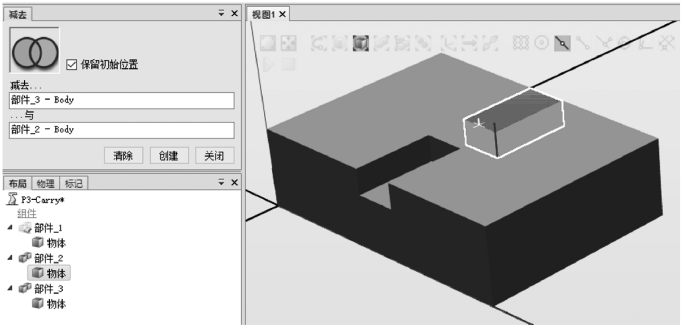
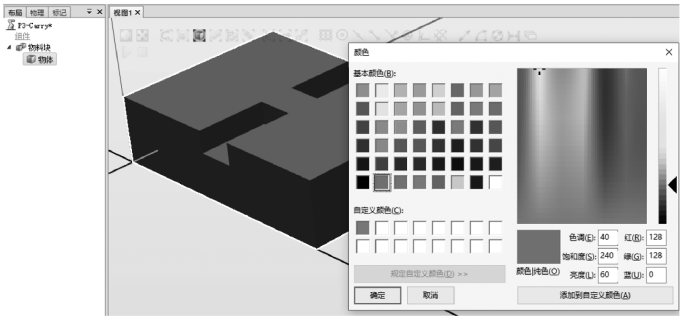
续表3-1

图片示例	步骤说明
	步骤 2：设置大矩形体的尺寸参数 说明：设置矩形体的长度、宽度、高度等参数。其中角点是指矩形体从该点创建，该点的位置参数基于“参考”中的坐标系；方向指与坐标系的相对方向，其值为角度。当前角点和方向均为默认，即在世界坐标系原点，且沿世界坐标系方向
	步骤 3：创建小矩形体 说明：采用同样的方式创建小矩形体，长度设置为 16 mm、宽度设置为 8 mm、高度设置为 5 mm。为了避免和大矩形体重叠，此处将角点坐标设置为(0，0，30)，即在原点的正上方 30 mm 处创建
	步骤 4：旋转小矩形体 说明：在视图中选中小矩形体，并右键选择“旋转”，在旋转参数中选中“Z”，并将旋转角度设置为 90°，单击“应用”。即将小矩形体绕 *Z* 轴旋转 90°，使其与大矩形体垂直

续表3-1

图片示例	步骤说明
	步骤 5：调整相对位置 说明：在视图中选中小矩形体并右击，依次选中“放置→一个点”。放置的方法有多种，此处由于两个矩形体处于平行状态，因此一点法即可实现放置要求。 通过捕捉中点，将“主点-从”设置为小矩形体左侧上表面中点，“主点-到”设置为大矩形体左侧上表面中点，最后点击“应用”，应用前确保小矩形体为选中状态。 注意：在捕捉点位时一定要先将光标定位至对应的点位数据框中，否则无法自动捕捉并填充点位数据
	步骤 6　切除左侧 说明：选择“建模→CAD 操作减去”，将“减去…”设置为大矩形体，“…与”设置为小矩形体，单击“创建”，即在大矩形体上切除了小矩形体部分。此时，会生成一个新的几何体，并自动命名为“部件_3”，如果将“部件_1”“部件_2”取消可见即可看到新生成的“部件_3”的形状

续表 3-1

图片示例	步骤说明
	步骤 7　切除右侧 说明：采用同样的方式将“部件_3”的右侧做切除得到新的几何体“部件_4”。为了放置时方便选取点位，可以将“部件_2”向上拖移后再进行放置，为了防止“部件_1”的干扰，可以将其删除或取消可见。此外，在进行减去操作时，“减去…”对应的值应设置为“部件_3”
	步骤 8　设置颜色 说明：在视图中选中最终生成的几何体，右键单击选择“设定颜色”，将颜色值设定为如左图所示。最后，在左侧树形菜单中右键单击“部件_4”，选择“重命名”，将“部件_4”重命名为“物料块”

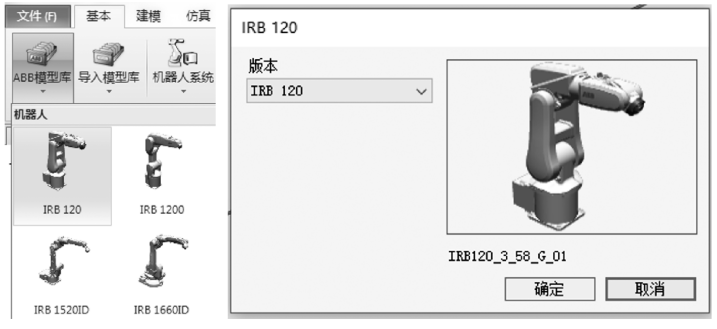
扫码观看工作台、工业机器人、搬运组件布局操作视频

3.1.3　工作站布局

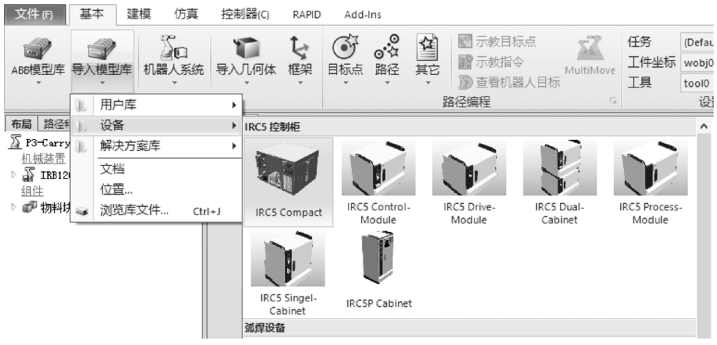
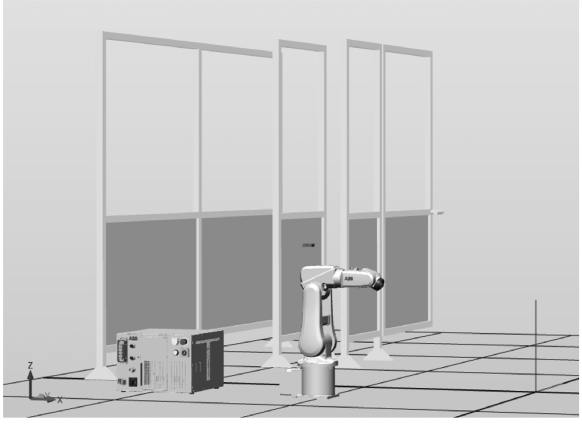
(1)模型库导入

RobotStudio 软件有着资源丰富的模型库，包括 ABB 各型号工业机器人、控制柜、工具、护栏、导轨等，根据此项目要求，需要从模型库中导入“IRB 120”工业机器人、IRC5 紧凑型控制柜、安全护栏，其导入过程如表 3-2 所示。

表 3-2　模型库导入操作步骤

图片示例	步骤说明
	步骤 1：导入工业机器人 说明：依次选择“基本→ABB 模型库”，在机器人下拉列表中选择“IRB 120”，然后在弹出的版本选择框中选择所需版本并单击“确定”，本项目中版本默认不修改

续表3-2

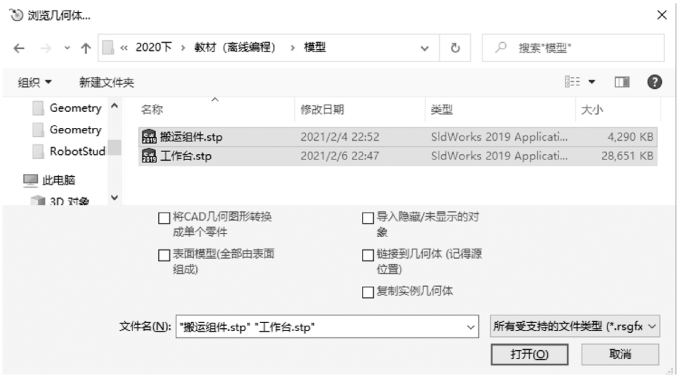
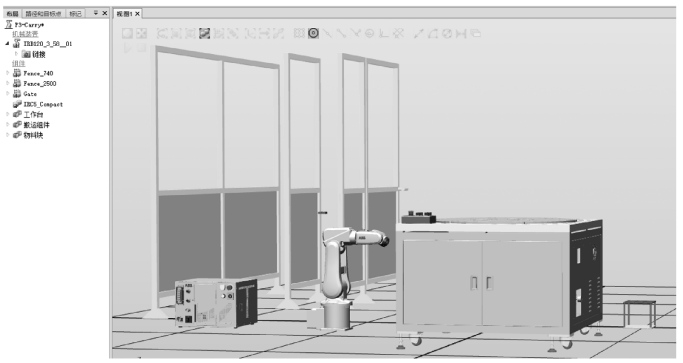
图片示例	步骤说明
	步骤 2：导入控制柜 说明：依次选择“基本→导入模型库→设备”，在右侧下拉列表的“IRC5 控制柜”栏中选择“IRC5 Compact”
	步骤 3：导入安全护栏 说明：依次选择“基本→导入模型库→设备”，在右侧下拉列表的其他栏中导入“Fence 2500”“Fence 740”与“Fence Gate”
	步骤 4：调整模型位置 说明：由于模型导入后会将模型的本地坐标原点与工作站中的大地坐标原点重合，因此导入的模型会重叠，为便于后续的模型导入与布局，可以将已导入模型拖移到合适位置

(2)几何体导入

对于相对复杂的个性化三维模型，通常需要在专业的三维建模软件中创建后再导入工作站中，本项目中需要导入已经创建好的工作台与搬运组件(物料盘)模型，其导入过程如表 3-3 所示。

扫码下载搬运组件模型

表 3-3 几何体导入操作步骤

图片示例	步骤说明
	步骤 1：浏览几何体 说明：依次选择“基本→导入几何体→浏览几何体”
	步骤 2：导入几何体 说明：在弹出的“浏览几何体”对话框中调整路径为几何体所在文件夹，框选“工作台”与“搬运组件”模型文件，最后单击“打开”
	步骤 3：调整位置 说明：在完成工作台与搬运组件的导入后，为了布局的方便可以先不移动工作台，将搬运组件在 X 或 Y 方向移动适当距离，使其不被工作台遮挡。此时，从左侧布局列表与视图中可以查看所有已经导入的组件，在列表中选中某一组件时，视图中会高亮显示

(3)工作站布局

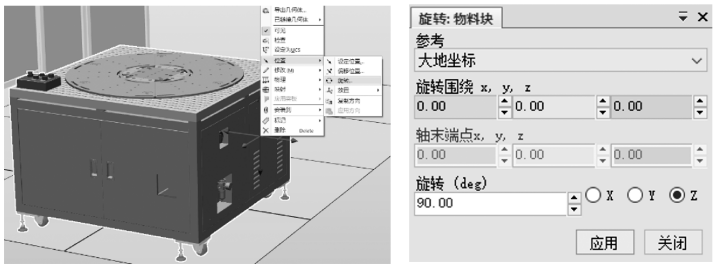
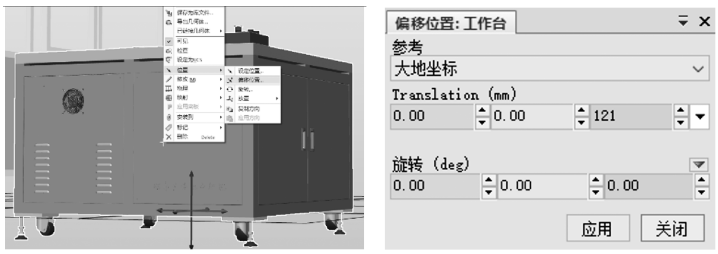
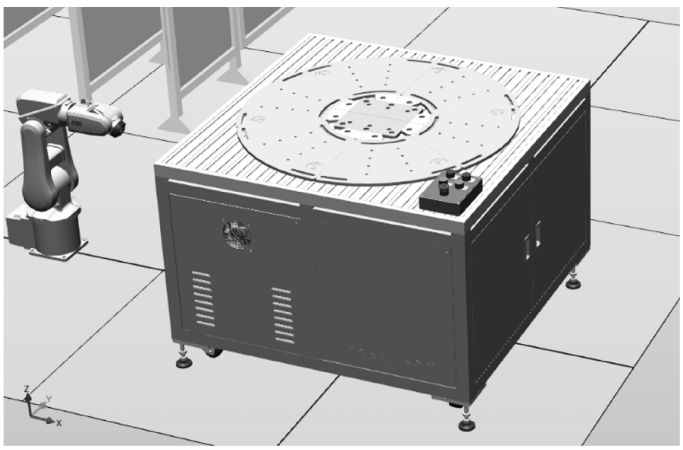
导入所需的设备与装置后，需要将导入的模型进行合理布局，使其满足搬运任务的工作要求，工作站布局主要包括工作台、工业机器人、搬运组件、物料块、安全护栏与控制柜等设备或组件的布局。

1)工作台

在实际的工业机器人工作站中，工作台是工作站的重要组成部分，工业机器人、示教器、PLC、触摸屏、搬运等功能组件以及设备之间的连接线路等都要安装于工作台上。通

常将工作台放置于地面中心位置，由于当前工作台朝向不正确，且导入时工作台的本地坐标原点(底部横梁下表面的中心)与工作站大地坐标原点重合，此时其四个承重脚处于地面以下，且承重脚下沿与地面垂直距离为 121 mm，因此需要进行旋转与偏移操作，具体操作如表 3-4 所示。

表 3-4　工作台布局操作步骤

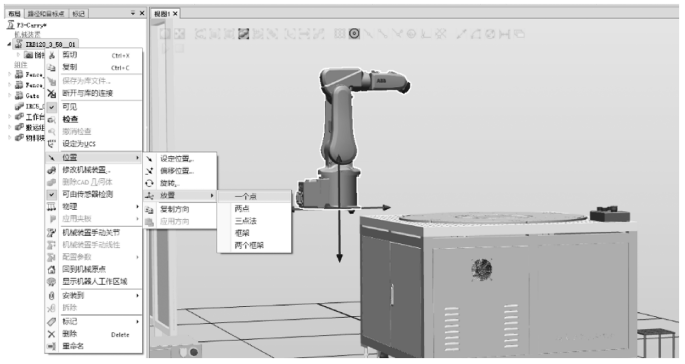
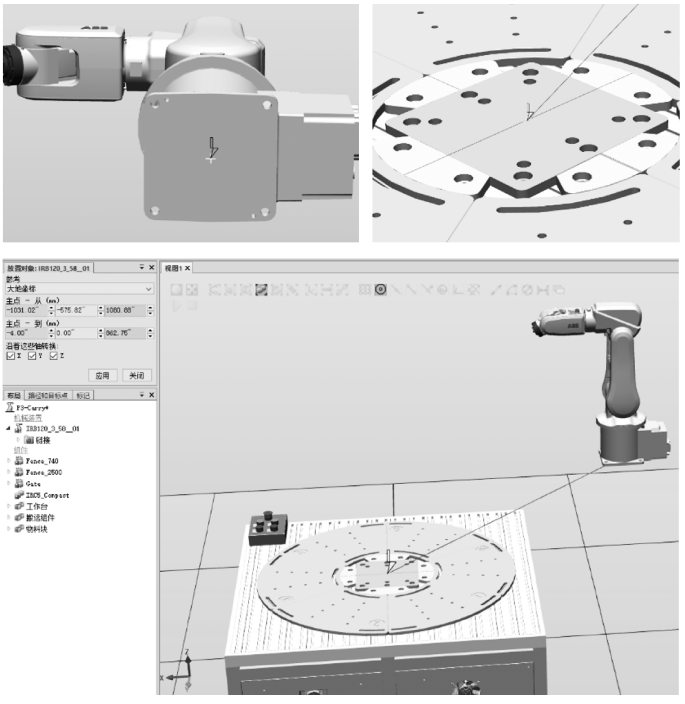
图片示例	步骤说明
	步骤 1：旋转工作台 说明：将工作台绕 Z 轴旋转 90°，使工作台 3 号安装工位朝前(沿 X 轴方向)。 操作：右键单击工作台，选择“位置→旋转”，参数如左图所示
	步骤 2：平移工作台 说明：将工作台向正上方平移 121 mm，使工作台放置于地面。 操作：右键单击工作台，选择“位置→偏移位置”即可进行偏移参数设置
	步骤 3：位置确认 说明：确认工作台的 3 号安装工位沿 X 轴方向且承重脚正好位于地面

在进行位置偏移时，“参考”是指偏移参考的坐标，可选大地坐标、父级、本地、UCS；“Translation”处所填值即为偏移值，三个值分别为对应参考坐标轴的 X、Y、Z 方向的偏移值，值为正表示向该轴正方向偏移，为负则表示沿该轴负方向；“旋转”处的值对应在偏移的同时绕 X、Y、Z 轴的旋转角度。在输入偏移值后视图中即可预览偏移效果，应用后偏移生效。

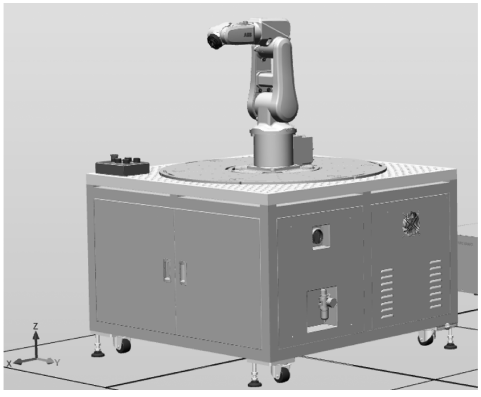
2)工业机器人

工业机器人是整个工作站的核心，工作台上设置了工业机器人安装位，因此需要将工业机器人安装于该位置上。工业机器人的布局操作步骤如表 3-5 所示。

表 3-5　工业机器人布局操作步骤

图片示例	步骤说明
	步骤 1：确认放置方法 说明：工业机器人与工作台朝向相同，且机器人底部与工作台上的机器人安装位平行，因此可以采用一点法放置。 操作：为了取点的方便，先将工业机器人向上方拖移一定位置，然后在布局列表中右键单击机器人，依次选择“位置→放置→一个点”
	步骤 2：点位捕捉 说明：将工业机器人底部中心放置到工作台机器人安装位中心。 操作：在弹出的放置对象参数设置窗口中，“主点-从”捕捉机器人底部中心位置，“主点-到”捕捉工件台上机器人安装位置的中心，最后单击“应用”

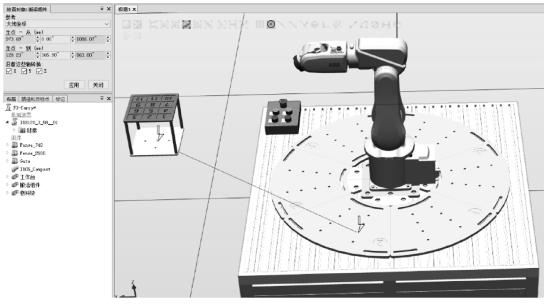
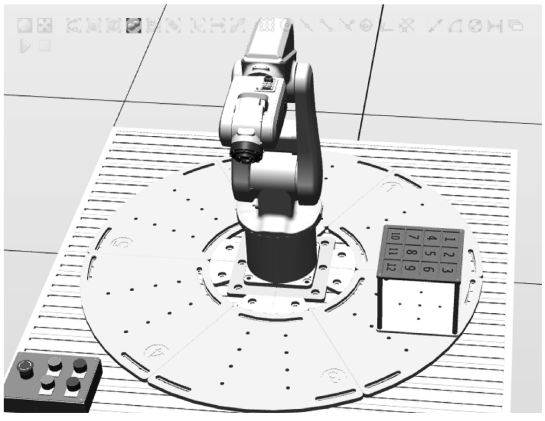
续表3-5

图片示例	步骤说明
	步骤 3：位置确认 说明：确认工业机器人正确放置于工作台对应的机器人安装位，且朝向与工作台一致

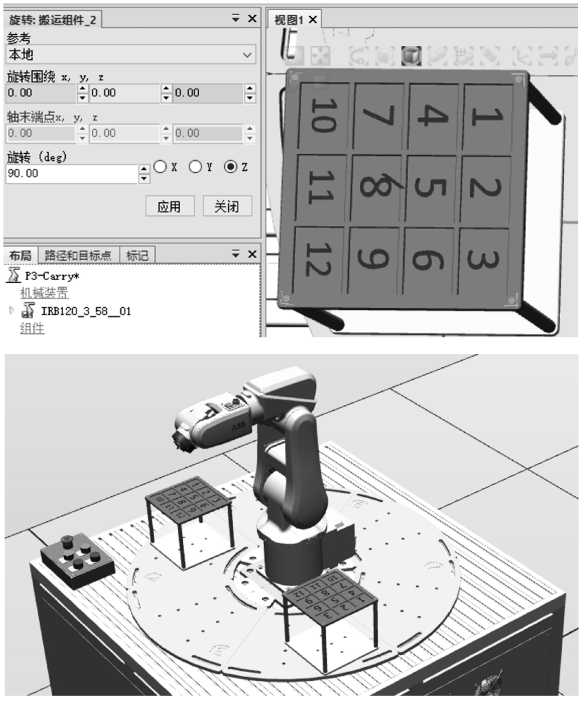
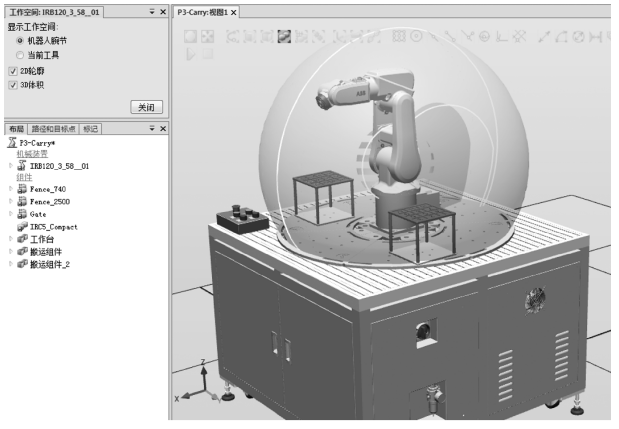
3）搬运组件

搬运组件，即物料盘，是安装于工作台的实现搬运任务的功能组件，用于放置物料块。本项目中需要将物料块从一个物料盘搬运至另一物料盘，最后再搬回至初始物料盘，因此，需要放置两个物料盘，分别安装于工作台 2 号与 4 号安装工位，具体操作步骤如表 3-6 所示。

表 3-6　工作台布局操作步骤

图片示例	步骤说明
 	步骤 1：安装物料盘 1 说明：在工作台的 2 号工位安装一个物料盘。 操作：先将物料盘向上方拖移一定距离，然后采用一点法进行放置，“主点-从”设置为物料盘的底部中心，“主点-到”设置为工作台上 2 号工位某一安装孔的中心，确认无误后单击“应用”

续表3-6

图片示例	步骤说明
	步骤 2：复制物料盘 说明：在工作台的 2 号工位复制并生成一个相同的物料盘。 操作：在布局列表中右键单击搬运组件，选择“复制”，然后在工作站名称处单击右键选择“粘贴”，此时会自动在相同位置生成一个名称为“搬运组件_2”的组件
	步骤 3：放置物料盘 2 说明：在工作台的 4 号工位安装一个物料盘并调整方向。 操作：选中“搬运组件_2”，拖移其 *X* 或 *Y* 坐标轴，将其移动至 4 号工位合适位置，然后右键单击“搬运组件_2”，选择“位置→旋转”，将“参考”设置为“本地”，旋转轴设置为“Z”，旋转角度设置为 90°，确认无误后单击“应用”，最终放置效果如左图(下)所示
	步骤 4：确认工作范围 说明：确认物料盘在工业机器人的工作范围之内。 操作：在布局面板中选中机器人并单击右键，选中“显示机器人工业区域”，在“工作空间”设置窗口中勾选“3D 体积”。如果物料盘在显示的 3D 范围内则无须再调整物料盘的位置。当然，在后续安装工具后其实际工作范围会有一定的变化

需要注意的是，在旋转操作时其旋转后的位置不仅取决于旋转轴与旋转角度，还与参考坐标有关，其可选参考坐标包括大地坐标、父级、本地、UCS、用户自定义。默认情况下，父级、UCS 与大地坐标相同，大地坐标原点在地面中心位置，坐标方向与视图左下角坐标系指向相同；本地则是指每个组件的本地坐标，当某一个组件被选中时，该组件会高亮并显示一个坐标系，该坐标系即为其本地坐标，选中该坐标系的某个坐标轴并移动即可拖移该组件。

扫码观看物料块、安全护栏与控制柜布局操作视频

4）物料块

物料块是搬运任务的操作对象，本项目中，物料盘共有 12 个物料位，任务要求将物料块从左侧（4 号工位）物料盘搬运至右侧（2 号工位）物料盘，因此，需要在左侧物料盘上的每一个物料位放置一个物料块。

由于物料块较多，一个一个放置会比较烦琐，观察物料盘布局可知，物料位置呈矩阵分布，4 行 3 列，因此可以先完成一个物料块放置，然后再复制和偏移。物料块的尺寸已知（长 55 mm，宽 40 mm），且物料位的尺寸与物料块相同，每个物料之间的间隔可以通过测量工具测得。物料盘放置物料块具体思路如图 3-4 所示。

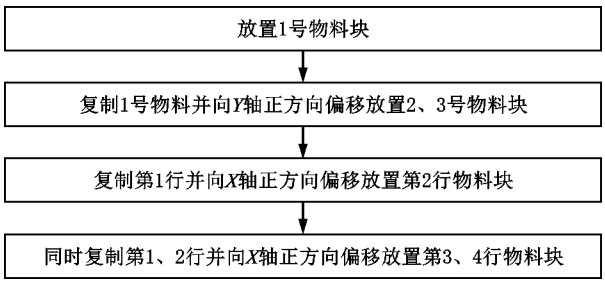
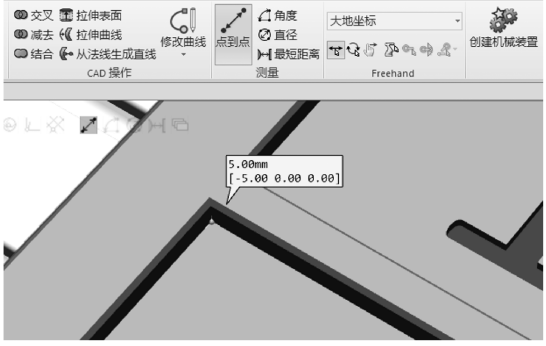


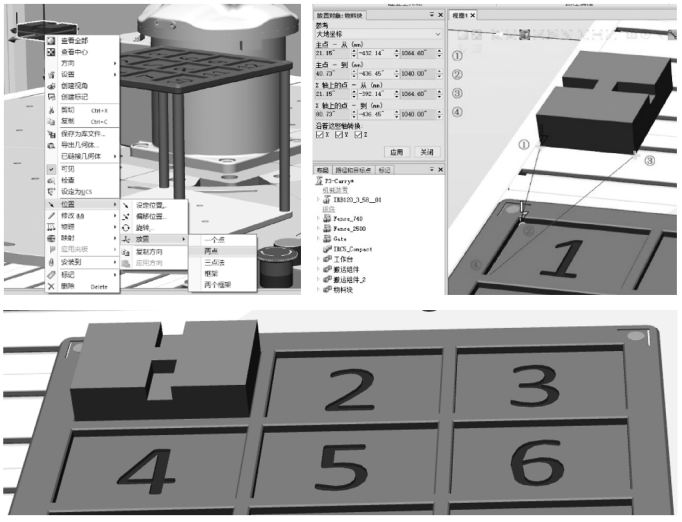
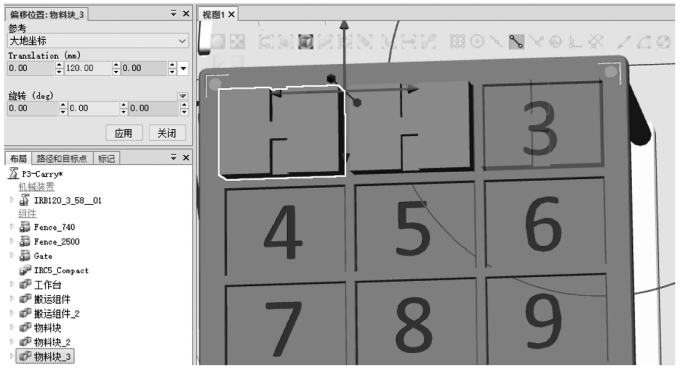
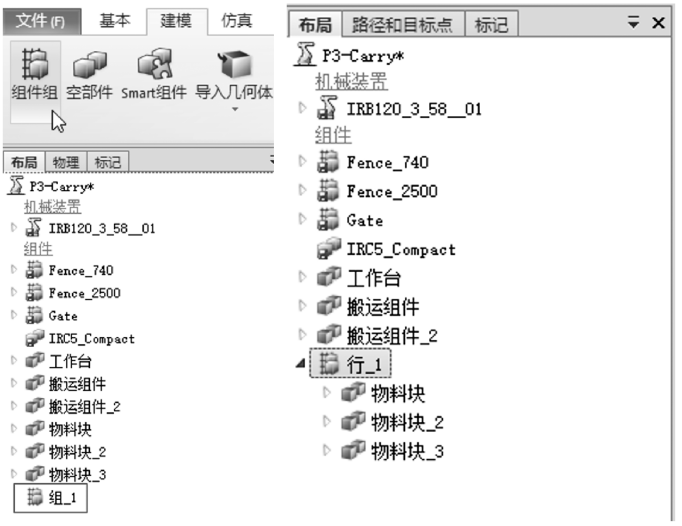
图 3-4　物料盘放置物料块基本思路

物料块布局具体操作步骤如表 3-7 所示。

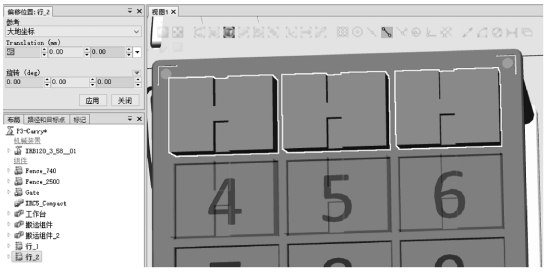
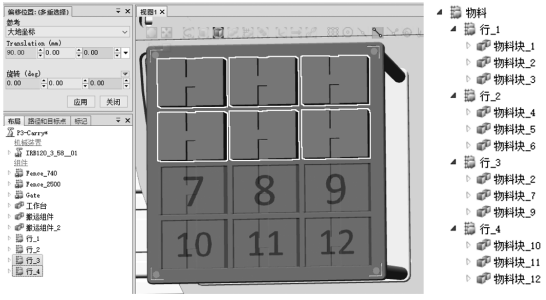
表 3-7　物料块布局操作步骤

图片示例	步骤说明
	步骤 1：测量物料块间隔 说明：测量物料位行方向与列方向的间隔。 操作：依次选中“建模→点到点”，然后激活捕捉末端，依次捕捉两相邻物料位凹槽中的相邻顶点，可测得行与列方向间隔均为 5 mm。 因此相邻两列需要偏移 60 mm，相邻两行偏移 45 mm

续表3-7

图片示例	步骤说明
	步骤 2：放置第 1 块物料 说明：将已经创建好的物料块放置于左侧物料盘的 1 号位置。 由于创建的物料块与物料位的朝向相互垂直，可采用两点法放置。 操作：将物料块拖移至物料盘上方适当位置，右击物料块，依次选择“位置→放置→两点”。激活捕捉末端，按如左图所示依次捕捉点位，确认无误后单击“应用”
	步骤 3：放置第 2、3 块物料 说明：将 1 号位物料块复制两次，并分别向 Y 轴正方向偏移 60 mm 与 120 mm，完成第一行物料块的放置
	步骤 4：创建组件组 说明：创建一个组件组，用于整理各物料块。 操作：依次选择“建模→组件组”，布局列表中会生成名称为“组_1”的组件组，将其重命名为“行_1”，然后选中第一行 3 个物料块并拖放至该组中

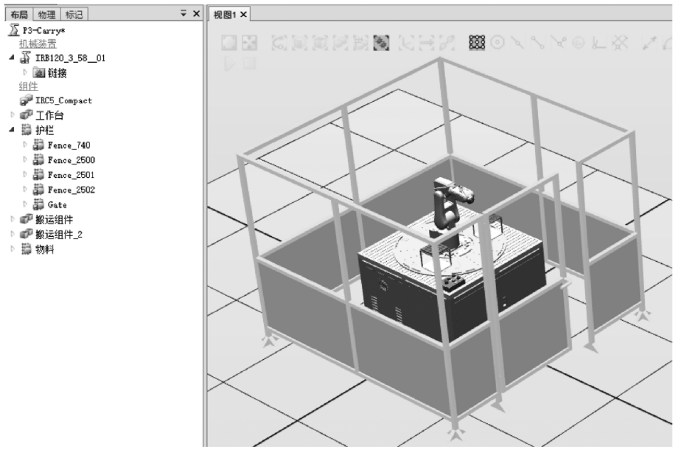
续表3-7

图片示例	步骤说明
	步骤 5：放置第 2 行 说明：复制组件组“行_1”，并将其向 X 方向偏移 45 mm
	步骤 6：放置第 3、4 行 说明：同时复制第 1、2 行，并将复制的两行同时向 X 方向偏移 90 mm。为了方便识别，可以将组件组与各物料块名称按编号重命名。最后，新建一个组件组并重命名为“物料”，将 4 行物料组都移入该组中

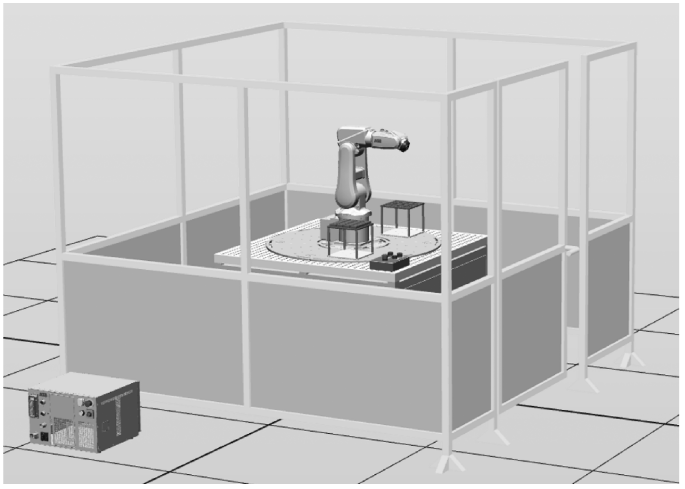
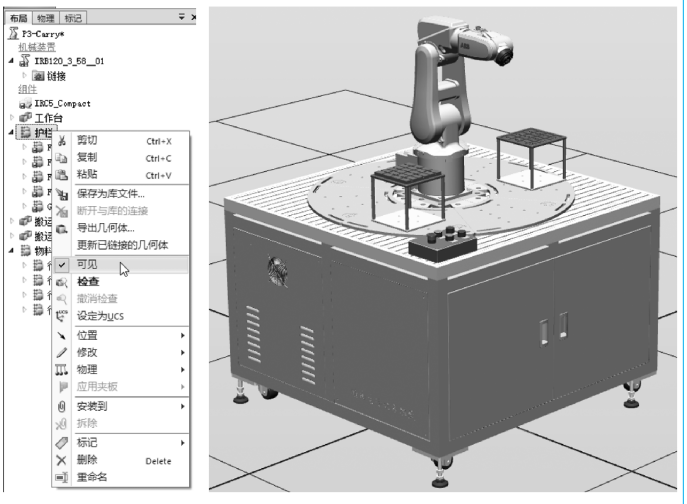
5）安全护栏与控制柜

在实际应用中，工业机器人工作时运行速度非常快，容易对靠近的人员造成人身伤害，因此，通常需要设置安全护栏与安全光栅等安全防护装置，将人员隔离在工业机器人的工作范围之外。此外，控制柜是工业机器人的重要组成部分，可以在工作站合适位置布局工业机器人控制柜，但是，控制柜模型的有无并不影响后续系统的创建。安全护栏与控制柜设置步骤如表 3-8 所示。

表 3-8　安全护栏与控制柜布局操作步骤

图片示例	步骤说明
	步骤 1：布局安全护栏 说明：在工作站四周设置安全护栏，设置合理即可。 操作：将“Fence_2500”组件放置于工作台后方合适位置，并复制两个“Fence_2500”组件分别设置于工作台左右两侧，然后将“Gate”“Fence_740”两个组件分别设置于工作站前方。最后，新建一个组件组并重命名为“护栏”，将安全护栏的所有组件移动至该组中，并根据实际情况整体移动“护栏”，使工作台置于护栏所围区域中间位置

续表3-8

图片示例	步骤说明
	步骤 2：设置控制柜 说明：将控制柜(IRC5_Compact)拖移至安全护栏外合适位置
	步骤 3：设置护栏与控制柜不可见 说明：为了后续操作的方便，将护栏与控制柜等装置设置为不可见，完成程序调试后再将其设置为可见。 操作：在布局列表中右键单击“护栏”，取消“可见”的选中状态，视图中护栏即会隐藏，同样的方法设置控制柜。当需要显示这些装置时，勾选“可见”即可

3.1.4 系统创建

在完成工业机器人工作站布局后，此时机器人并不能直接编程运行，还需要为工业机器人创建系统，为其建立虚拟控制器，从而让其具备电气属性。为机器人创建系统的方法有三种，分别为“从布局”“新建系统”“已有系统”。

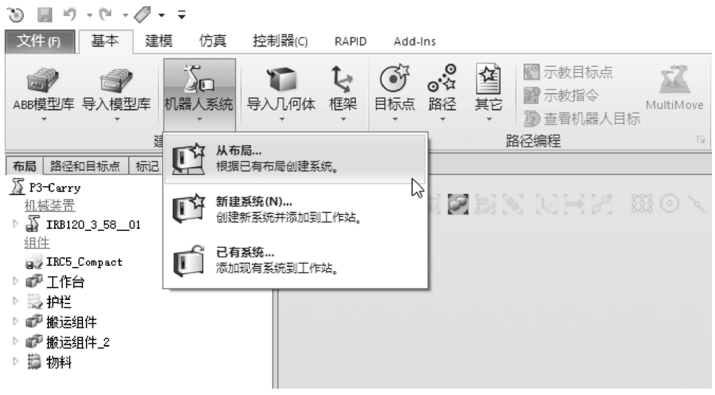
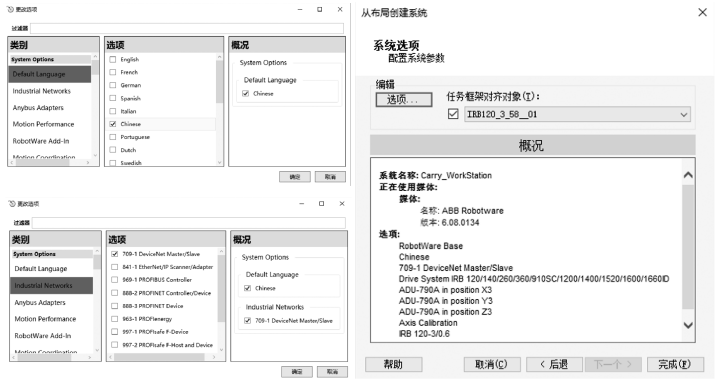
①“从布局”创建系统：根据当前的布局创建系统。完成布局后创建系统通常使用这种方法，这也是比较常用的方法。

②“新建系统”：选择机器人型号新建系统，新建系统后会自动导入该机器人到系统中，也可以从已有备份创建系统。

③“已有系统”：添加现有的系统到工作站中。

本项目已经完成了工作站布局，因此，选用“从布局”创建系统，具体操作步骤如表 3-9 所示。

表 3-9　从布局创建系统操作步骤

图片示例	步骤说明
	步骤 1：选择系统创建方式 说明：选择“从布局”创建系统，开启系统创建向导。 操作：依次选择“基本→机器人系统→从布局”
	步骤 2：设置系统参数 说明：设置好系统名称、位置、RobotWare 版本、机械装置。 操作：在弹出的“从布局创建系统”对话框中设置系统名称为“Carry_WorkStation”，选择保存位置，当有多个版本的 RobotWare 时选择所需的版本，单击“下一个”，机械装置选择默认值，单击“下一个”
	步骤 3：设置系统选项 说明：将系统语言设置为“中文”，通信方式设置为工业网络“709-1 DeviceNet Master/Slave”。 操作：单击“选项”，在弹出的“更改选项”对话框中选择“类别”中的“Default Language”，勾选“Chinese”并取消勾选“English”；在“类别”中选择“Industrial NetWorks”，勾选“709-1 DeviceNet Master/Slave”，最后单击“确定”，在“概况”信息栏中确认系统参数无误后单击“完成”

续表3-9

图片示例	步骤说明
	步骤 4：系统创建完成 说明：当控制器状态显示绿色，且状态为“已启动”，表示系统创建完成并且已经正常工作

扫码观看夹具工具创建操作视频

任务 2　夹具工具创建

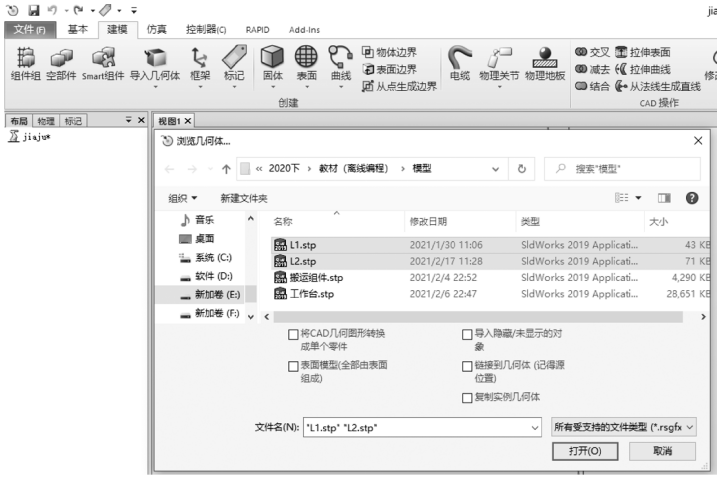
夹具的创建可以直接在已经布局好的工作站中创建，也可以新建空工作站后创建，然后保存为库文件并导入已经完成布局的工作站中。本任务采用第二种方式，新建名称为“jiaju”的空工作站。

扫码下载夹具模型

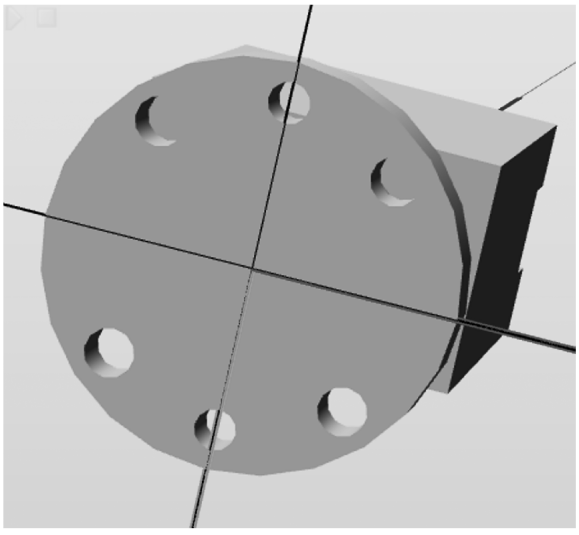
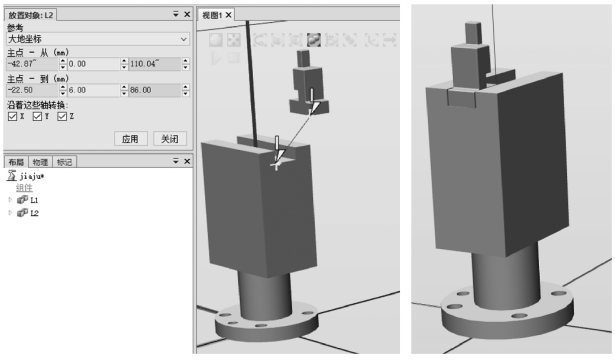
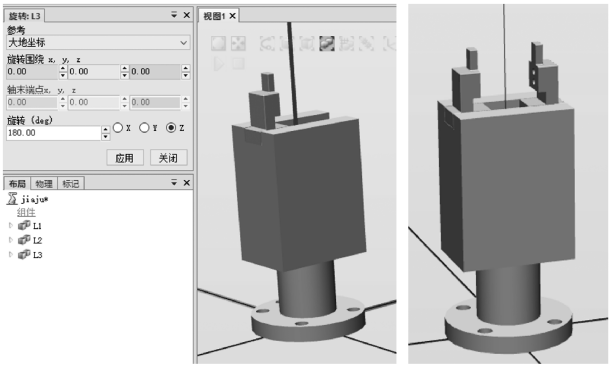
3.2.1　夹爪安装

夹具模型包含夹具座与夹爪两个部分，其导入与位置调整过程如表 3-10 所示。

表 3-10　夹具模型导入与夹爪安装操作步骤

图片示例	步骤说明
	步骤 1：导入夹具模型 说明：新建工作站后，通过导入几何体方式导入夹具模型“L1.stp”(夹具座)与“L2.stp”(夹爪)

续表3-10

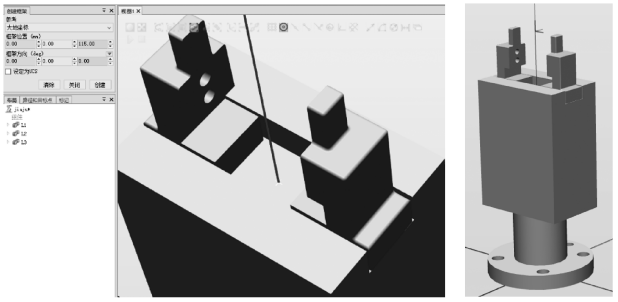
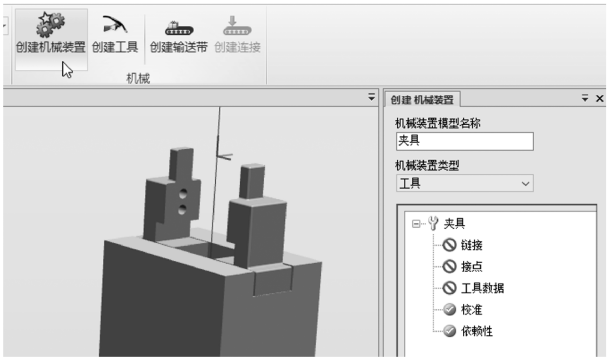
图片示例	步骤说明
	步骤 2：设置本地原点 说明：工具安装时是将工具的本地坐标与机器人法兰盘的 Tool0 坐标重合，且创建工具时，系统会自动将大地坐标原点设置为工具的本地坐标原点，因此，需要将夹具安装法兰中心位置放置于大地坐标原点。本任务中，由于夹具座的本地坐标原点已经在安装法兰中心，导入后即会与大地坐标重合，因此无须再进行设置，保持夹具座不移动即可
	步骤 3：安装夹爪 1 说明：保持夹具座在大地坐标原点不动，拖移与旋转夹爪，使夹爪安装部位与夹具座上的安装槽平行，并通过一点法放置夹爪。由于安装槽比夹爪安装部位宽 1 mm，因此需要向中间偏移 0.5 mm
	步骤 4：安装夹爪 2 说明：复制已经放置好的夹爪，并重命名为“L3”，将其绕 Z 轴旋转 180°

3.2.2　机械装置(工具)创建

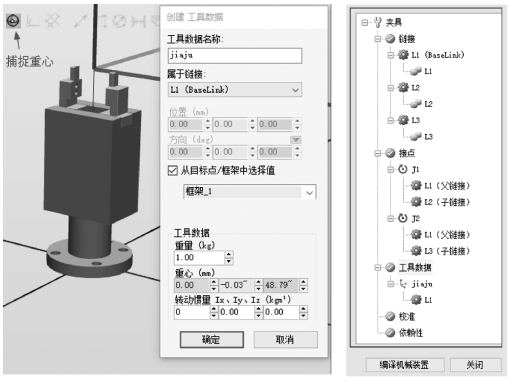
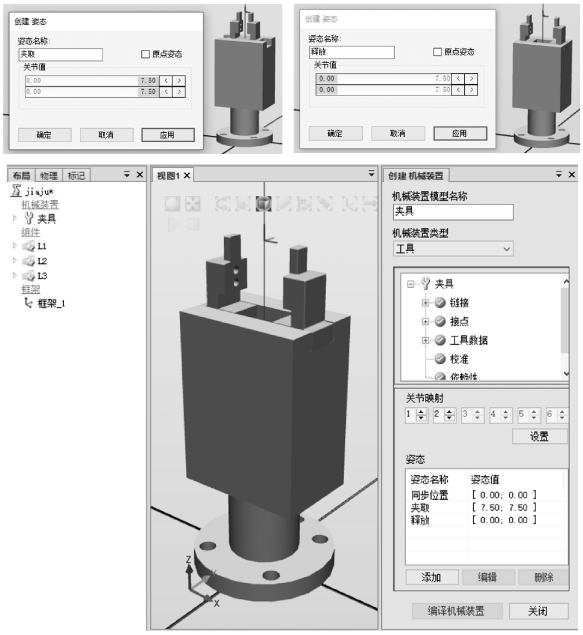
工具具备运动关节后才能完成夹紧、松开等动作，前述操作中只是将独立的夹爪安装在了夹具座上，但并不具备运动关节，因此，需要通过创建类型为工具的机械装置，为其设置

运动关节与动态姿态，并创建工具数据。机械装置(工具)创建过程如表 3-11 所示。

表 3-11　机械装置(工具)操作步骤

图片示例	步骤说明
	步骤 1：设置工具坐标框架 说明：在两个夹爪的顶端中心位置创建坐标框架，用于创建工具数据。 操作：依次选择“基本→框架→创建框架”，框架位置捕捉夹具座凹槽中心，由于夹爪高 29 mm，所以将捕捉到的位置数据(0，0，86)的 Z 轴加 29，框架方向不修改，确认无误后单击“创建”
	步骤 2：创建机械装置 说明：创建“工具”类型的机械装置。 操作：依次选择“建模→创建机械装置”，“机械装置模型名称”设置为“夹具”，“机械装置类型”设置为“工具”
	步骤 3：设置链接 说明：基于 L1、L2、L3 三个组件创建三个链接，其中 L1 设置为 BaseLink。 操作：双击“链接”，在弹出的“创建链接”窗口中设置“链接名称”为“L1”，“所选组件”选择“L1”并勾选“设置为 BaseLink”，单击右三角按钮 ▶，然后单击“应用”；接着修改“链接名称”为“L2”，“所选组件”选择“L2”，单击右三角按钮，单击“应用”；同样的方式将组件“L3”设置为链接“L3”，最后单击“确定”关闭窗口

续表3-11

图片示例	步骤说明
	步骤 4：设置接点 说明：接点用于连接各链接，相当于关节，通过设置接点，使两个夹爪在夹具座凹槽中往复移动，实现夹紧和释放动作。 操作：双击“接点”，在弹出的“创建接点”对话框中，“关节名称”设置为“J1”，“关节类型”选择“往复的”，“父链接”与“子链接”分别为“L1”“L2”，“关节轴”第二个位置的 X 值设置为“1”，“关节限值”中最小限值设为“0”，最大限值设为“7.5”，此时，可以拖动“操纵杆”预览夹爪的动作效果，确认无误后单击“应用”，同样的方式设置关节“J2”，参数如左图所示
	步骤 5：设置工具数据 说明：创建夹具的工具坐标数据。 操作：双击“工具数据”，在弹出的“创建工具数据”窗口中，设置“工具数据名称”为“夹具”，“属于链接”设置为“L1”，“位置”从框架中获取，勾选“从目标点/框架中选择值”，将光标定位到输入框中后，在布局列表中选择步骤 1 中创建好的框架，“重心”可以捕捉“L1”的重心，重量与转动惯量不修改，确认无误后单击“确定”
	步骤 6：编译机械装置并添加姿态 说明：当前述步骤操作完成后，“编译机械装置”按钮会激活，编译机械装置后再设置“夹取”与“释放”两个姿态。 操作：单击“编译机械装置”，此时布局列表中会出现名称为“夹具”的工具；单击“姿态”下方的“添加”，在弹出的“创建姿态”对话框中，设置“姿态名称”为“夹取”，“关节值”均设置为“7.5”，单击“应用”；然后设置“释放”姿态，“关节值”均为“0”，设置完成后，在“姿态”窗口中选中设置好的姿态，视图中即可看到姿态的效果。此外，也可以通过“手动关节”或“机械装置手动关节”控制两个夹爪单独运动。 注意：为了避免几何体模型对查看姿态效果的影响，可以将组件“L1”“L2”“L3”设置为不可见或删除

注意：工具的安装原理为工具的本地坐标系与机器人法兰盘坐标系Tool0重合，工具的工具坐标系框架作为机器人的工具坐标系。在创建工具时，系统会默认将大地坐标原点设定为工具的本地坐标原点，因此，为确保创建好的工具能正确安装到机器人第6轴中心，在创建工具前通常还需要设置其本地原点，即将工具的安装法兰中心放置于大地坐标原点，且保持正确姿态。由于夹具安装座的本地坐标原点已经在安装法兰位置的中心，且导入后未移动其位置，即保持夹具的安装法兰中心与本地坐标原点重合，因此工具创建完成后其本地坐标原点正好在工具安装法兰的中心。

扫码观看夹具动态属性创建的操作视频

任务3　夹具动态属性创建

夹具机械装置创建完成后，其已经具有两个活动关节，能实现“夹取”与“释放”动作，但是其并不能真正夹取东西。夹具夹取和释放的动态仿真效果是实现搬运任务的关键步骤，其仿真效果的实现要基于Smart组件。

3.3.1　Smart组件介绍

Smart组件是RobotStudio仿真软件实现动态仿真效果的重要工具，通过该组件可以对机械装置、工具等进行控制，使机械装置与工具实现动态传送、拾取与释放物料等动态效果，且该组件动作可以由代码或/和其他Smart组件控制执行。

基于Smart组件创建夹具夹取与释放的动态属性需要用到表3-12所示子对象组件。

表3-12　创建动态夹具需用到的Smart组件的子对象组件

子对象组件	作用
LineSenSor	线传感器：用于检测要夹取的对象
PoseMover	姿态移动：机械装置关节运动到一个已定义的姿态
Attacher	安装一个对象：将子对象安装到父对象(夹取)
Detacher	拆除一个已安装的对象(释放)
LogicGate	逻辑信号(包括与或非，此处使用逻辑非)
LogicSRLatch	固定信号(夹具夹取对象后在释放前保持夹取状态)

3.3.2　利用Smart组件创建夹具动态属性

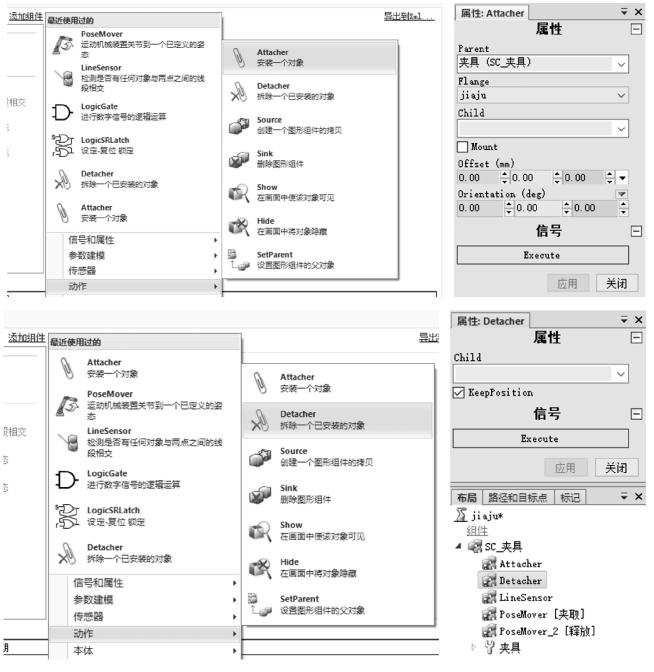
Smart组件创建的夹具的工件原理为：在夹具的夹爪中间安装一个传感器，当夹具执行夹取操作时，开启传感器，若传感器检测到夹爪中间有可拾取对象，则将该对象安装于夹具上，并输出夹取成功信号，用于程序控制；当夹具执行释放操作时则拆除已经被安装的对象并复位夹取成功信号。由

于传感器需要安装于“夹具”上，夹具在工作时可能会被传感器检测，而传感器只能检测一个对象，因此为了避免传感器检测到夹具而不能正常识别夹取对象，需要将“夹具”设置为不可被传感器检测。利用 Smart 组件创建夹具动态属性的操作如表 3-13 所示。

表 3-13　利用 Smart 组件创建夹具动态属性的操作步骤

图片示例	步骤说明
	步骤 1：新建 Smart 组件 操作：依次选择“建模→Smart 组件”，将新创建的 Smart 组件“SmartComponent_1”重命名为“SC_夹具”
	步骤 2：设置夹具为 Role 说明：将夹具设置为 Smart 组件的 Role，使“SC_夹具”继承夹具的工具坐标等属性。 操作：在布局面板中，选中夹具并拖动至“SC_夹具”后释放，此时，Smart 组件的组成中会出现夹具，右键单击“夹具”并勾选“设定为 Role”
	步骤 3：添加线传感器 说明：夹具夹取物料时利用该传感器识别被夹取对象。 操作：依次选择“添加组件→传感器→LineSensor”，在弹出的 LineSensor 属性框中，Start(起点)捕捉夹具座凹槽的中心，End(结束点)捕捉工具坐标原点，Radius(感应半径)设置为 1，应用后即可在视图中看到圆柱状的传感器。然后在“布局”面板中右键单击“夹具”，取消勾选“可由传感器检测”

续表3-13

图片示例	步骤说明
	步骤4：添加姿态运动子组件 说明：姿态运动子组件控制夹具夹爪在“夹取”与“释放”两个状态间切换。 操作：依次选择“添加组件→本体→PoseMover”，在弹出的PoseMover属性框中，“Mechanism”选择“夹具”，“Pose”选择“夹取”，“Duration(s)”设置为0.5，确认无误后单击“应用”；同样的方法继续添加一个PoseMover子组件，“Pose”设置为“释放”
	步骤5：添加安装与拆除子组件 说明：夹具夹取物料实际是将物料安装于夹具上，释放物料则为拆除。 操作：依次选择“添加组件→动作→Attacher”，在弹出的Attacher属性框中，“Parent”选择“夹具”，“Flange”自动填充“jiaju”，然后单击“应用”；接着依次选择“添加组件→动作→Detacher”添加“Detacher”子组件，Detacher组件属性无须修改，Attacher与Detacher组件中“Child”属性是安装与拆除的对象，其值通过属性连接设置

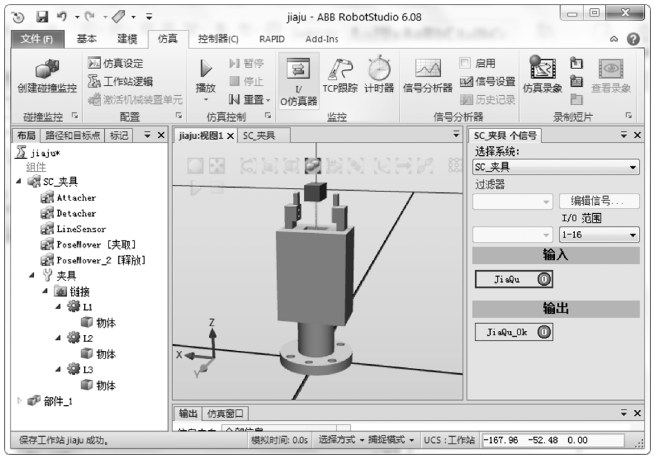
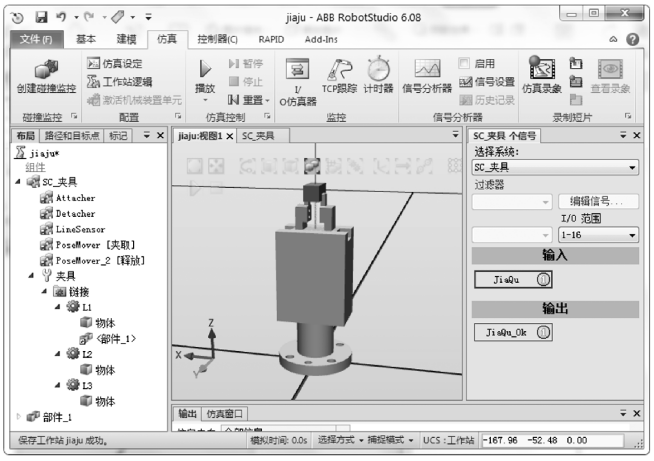
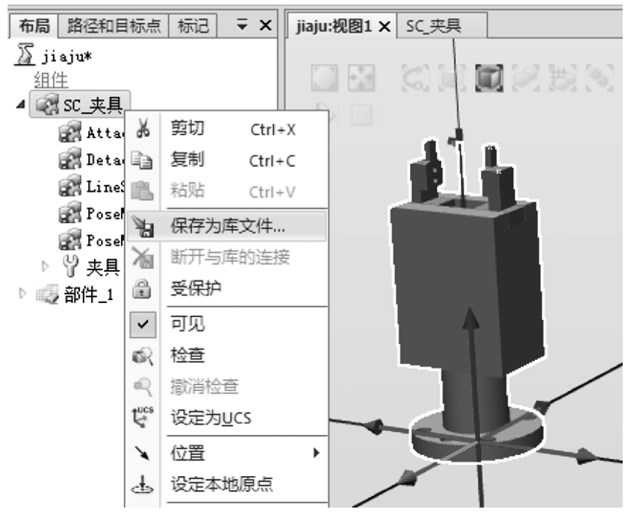
续表3-13

图片示例	步骤说明
	步骤 6：添加逻辑信号 说明：添加逻辑“非”门与 LogicSRLatch，LogicSRLatch 组件在夹取对象时进行锁定、释放对象时复位。 操作：依次选择“添加组件→信号和属性→LogicGate”，在弹出的属性框中，“Operator”选择“NOT”并应用；接着依次选择“添加组件→信号和属性→LogicSRLatch”，其属性无须设置。添加完成后，“SC_夹具”的子对象组件共包括一个“角色”，七个“Smart 组件”
	步骤 7：设置属性连接 说明：通过属性连接，将传感器检测到的对象作为 Attacher 组件的安装子对象，Attacher 组件安装的子对象又作为 Detacher 组件拆除的子对象。 操作：切换到“SC_夹具”的“属性与连结”，单击“属性连结”列表下方的“添加连结”，在弹出的“添加连结”对话框中设置如左图(上 1)，同样的方式添加如左图(上 2)的属性连接，在属性连结列表中可以查看已经添加的属性连结

续表3-13

图片示例	步骤说明
	步骤 8：添加 I/O 信号 说明：添加控制夹具和显示夹具状态的 I/O 信号。 操作：在“SC_夹具”编辑界面选择“信号和连接”，单击 I/O 信号列表下方的“添加 I/O Signals”，在弹出的对话框中，“信号类型”选择“DigitalInput”，“信号名称”输入“JiaQu”，如左图(上 1)，确认无误后单击“确定”；相同的方式添加名称为“JiaQu_Ok”、类型为“DigitalOutput”的信号，如左图(上 2)所示，I/O 信号列表中可以查看已经添加的信号，如左图(下)所示，各信号的作用如表 3-14 所示
 I/O连接 源对象 | 源信号 | 目标对象 | 目标信号或属性 SC_夹具 | JiaQu | LineSensor | Active SC_夹具 | JiaQu | PoseMover [夹取] | Execute LineSensor | SensorOut | Attacher | Execute Attacher | Executed | LogicSRLatch | Set LogicSRLatch | Output | SC_夹具 | JiaQu_Ok SC_夹具 | JiaQu | LogicGate [NOT] | InputA LogicGate [NOT] | Output | PoseMover_2 [释放] | Execute LogicGate [NOT] | Output | Detacher | Execute Detacher | Executed | LogicSRLatch | Reset 添加I/O Connection 编辑 删除 上移 下移	步骤 9：添加 I/O 连接 说明：关联各子组件信号，使其能实现夹取与释放功能。 操作：在“SC_夹具”编辑界面选择“信号和连接”，单击 I/O 连接列表下方的“添加 I/O Connection”，在弹出的对话框中，“源对象”选择“SC_夹具”，“源信号”选择“JiaQu”，“目标对象”选择“LineSensor”，“目标信号或属性”选择“Active”，如左图(上)所示；相同的方式依次添加如左图(下)共 9 条 I/O 连接。各 I/O 连接的含义如表 3-15 所示

续表3-13

图片示例	步骤说明
 	步骤 10：测试夹具功能 说明：测试基于 Smart 组件的夹具能否正常夹取和释放对象。 操作：新建一合适大小（如 15 mm×10 mm×10 mm）的矩形体，拖移至夹爪中间，使其能被传感器触及，如左图（上）所示。依次选择“仿真→I/O 仿真器”，在右侧面板中将选择系统切换至“SC_夹具”，此时输入信号“JiaQu”与输出信号“JiaQu_Ok”值均为 0。单击输入信号“JiaQu”，如果“JiaQu”与“JiaQu_Ok”信号均变为 1，视图中夹具夹爪夹紧，且布局列表中“SC_夹具”中“夹具”的“L1”链接下出现了所夹取的对象（此处为“部件_1”），或者使用移动工具移动“SC_夹具”时所夹取的对象跟着移动，则说明已经成功夹取对象，如左图（下）所示。然后单击“JiaQu”，将其置 0，如夹爪能松开并释放物料，则表示夹爪动态功能完成
	步骤 11：导出夹具 说明：将夹具导出为库文件，方便导入至已经完成布局的工作站中。 操作：在“布局”面板中右键单击“SC_夹具”，选择“保存为库文件”，在弹出的“另存为”对话框中选择适当的路径保存即可

续表3-13

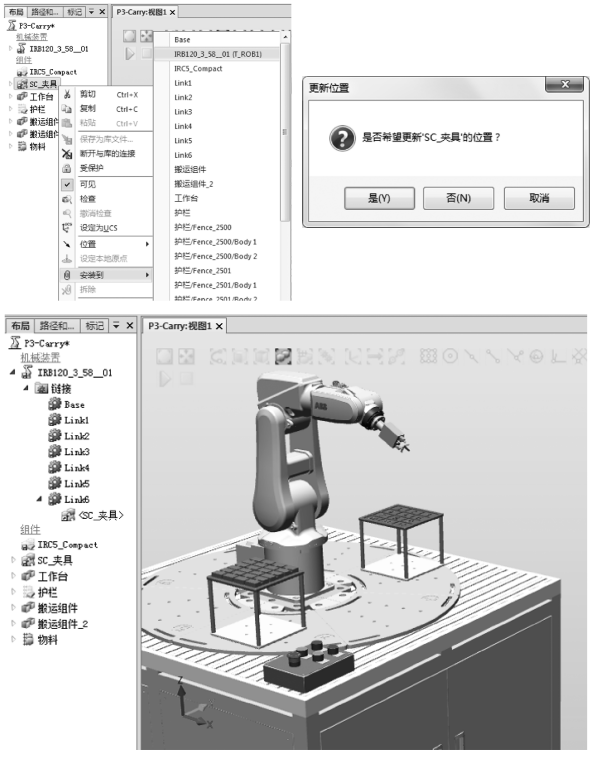
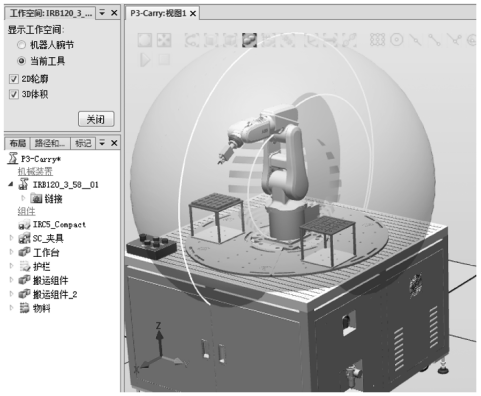
图片示例	步骤说明
	步骤 12：导入夹具 说明：将夹具导入已经完成布局的工作站中。 操作：打开已经布局完成的“P3-Carry”工作站，依次选择“基本→导入模型库→浏览库文件”，在弹出的“打开”对话框中定位至夹具存储位置，选择“SC_夹具. rslib”并单击“打开”
	步骤 13：安装夹具 说明：将夹具安装于工业机器人第 6 轴法兰盘。 操作： 方法 1：在“布局”面板中右键单击“SC_夹具”，依次选择“安装到→IRB120_3_58__01”，如左图(上 1)所示，在弹出的“更新位置”对话框中选择“是”，如左图(上 2)所示； 方法 2：在“布局”面板中选中“SC_夹具”并按下鼠标左键不松，将其拖至工业机器人上后松开鼠标。 安装完成后的效果如左图(下)所示
	步骤 14：工作范围确认 说明：确认物料盘是否在机器人工作范围内。 操作：在“布局”面板中右键单击机器人，选中“显示机器人工业区域”，在“工作空间”设置窗口中，“显示工作空间”选择“当前工具”，并勾选“3D 体积”，浅色阴影区域即为当前工具可到达的工作范围，如果物料块超出了工作范围则需要适当调整物料盘与物料块的位置，在范围内则无须调整

表 3-14　夹具 Smart 组件中各 I/O 信号的作用

I/O 信号	信号类型	值	作用
JiaQu	数字输入	1	夹具夹取
		0	夹具松开
JiaQu_Ok	数字输出	1	夹具夹到物料
		0	夹具未夹取或夹取但未夹到物料

表 3-15　夹具 Smart 组件中各 I/O 连接的含义

源对象	源信号	目标对象	目标信号或属性	含义
SC_夹具	JiaQu	LineSensor	Active	夹具的夹取信号被置 1 时激活传感器
SC_夹具	JiaQu	PoseMover［夹取］	Execute	夹具的夹取信号被置 1 时执行夹紧动作
LineSensor	SensorOut	Attacher	Execute	传感器检测到拾取对象则执行安装操作
Attacher	Executed	LogicSRLatch	Set	安装操作执行后固定信号保持夹取状态
LogicSrLatch	Output	SC_夹具	JiaQu_Ok	信号固定后将夹具夹取成功指示信号置 1
SC_夹具	Jiaqu	LogicGate［NOT］	InputA	将夹具的夹取信号取反，即为释放触发信号
LogicGate［NOT］	Output	PoseMover_2［释放］	Execute	释放信号被置 1 时夹具执行松开动作
LogicGate［NOT］	Output	Detacher	Execute	释放信号被置 1 时拆除已安装的对象
Detacher	Executed	LogicSRLatch	Reset	拆除动作执行后解除信号固定

任务 4　工作站逻辑设定

扫码观看工作站逻辑设定操作视频

通过 Smart 组件创建的具有动态属性的夹具可以由其 I/O 信号控制夹具的夹取与释放操作，同时还可以监控其是否成功夹取对象，但是工业机器人程序不能直接控制 Smart 组件中的 I/O 信号，需要在机器人控制器中创建 I/O 信号并与 Smart 组件中的信号进行关联，从而实现程序对夹具等周边设备的控制，也就是工作站逻辑设定。

3.4.1 控制器 I/O 信号创建

本项目中需要创建两个 I/O 信号，其中一个数字输出信号用于控制夹具的夹取与释放，一个数字输入信号用于监控夹具是否夹取成功，新建或编辑信号后必须重启控制器，否则更改不会生效，具体操作步骤如表 3-16 所示。

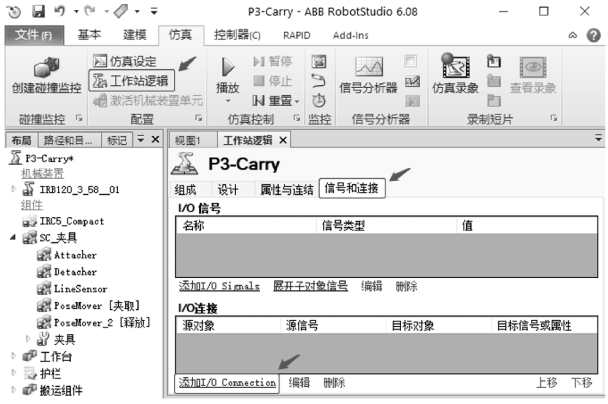
表 3-16 创建控制器 I/O 信号的操作步骤

图片示例	步骤说明
	步骤 1：打开 I/O System 操作：依次选择“控制器→配置→I/O System”
	步骤 2 新建 I/O 信号 说明：新建数字输入信号“Di_JiaQuOk”与数字输出信号“Do_JiaQu”。 操作：在“配置-I/O System”面板中，右键单击“Signal”，选择“新建 Signal”，在弹出的“实例编程器”对话框中将“名称”设置为“Di_JiaQuOk”，“Type of Signal”选择“Digital Input”，确认无误后单击“确定”，同样的方式创建名称为“Do_JiaQu”、类型为“Digital Output”的信号
	步骤 3：重启控制器 说明：重启控制器，使新建的 I/O 信号生效。 操作：依次选择“控制器→重启→重启动(热启动)”，在弹出的对话框中单击“确定”

3.4.2　工作站信号逻辑关联

创建控制器信号后，其与工作站中 smart 组件的信号是相互独立的，而程序中只能控制控制器中的信号，如果要实现程序对夹具的控制，则需要将控制器中的信号与工作站中 Smart 组件的信号进行关联，具体关联操作如表 3-17 所示。

表 3-17　工作站信号逻辑关联操作步骤

图片示例	步骤说明
	步骤 1：打开工作站逻辑设置面板 操作：依次选择“仿真→工作站逻辑”，在“工作站逻辑”设置面板中选择“信号和连接”，单击“I/O 连接”列表下方的“添加 I/O Connection”
  **I/O连接** 源对象 | 源信号 | 目标对象 | 目标信号或属性 Carry_WorkSta... | Do_JiaQu | SC_夹具 | JiaQu SC_夹具 | JiaQu_Ok | Carry_WorkSta... | Di_JiaQuOk	步骤 2：信号关联 说明：关联机器人系统的“Do_jiaQu”与“SC_夹具”的“JiaQu”信号。 操作：在弹出的“添加 I/O Connection”对话框中，“源对象”选择“Carry_WorkStation”，“源信号”选择“Do_JiaQu”，“目标对象”选择“SC_夹具”，“目标信号或属性”选择“JiaQu”，如左图(上)所示，确认无误后单击“确定”。同样的方式添加如左图(中)所示的信号关联，添加完成后 I/O 连接列表如左图(下)所示

任务5　搬运1个

编写搬运程序前，先要确定搬运顺序并对程序进行设计。本项目中依次从取料盘上按1→12的顺序夹取物料，并放置于放料盘上相同序号的位置上。在编程过程中，为了调试方便，可以先实现一个物料块的搬运，然后再完成一行，最后实现整盘的搬运，如图3-5所示。本任务先完成一个物料块的搬运。

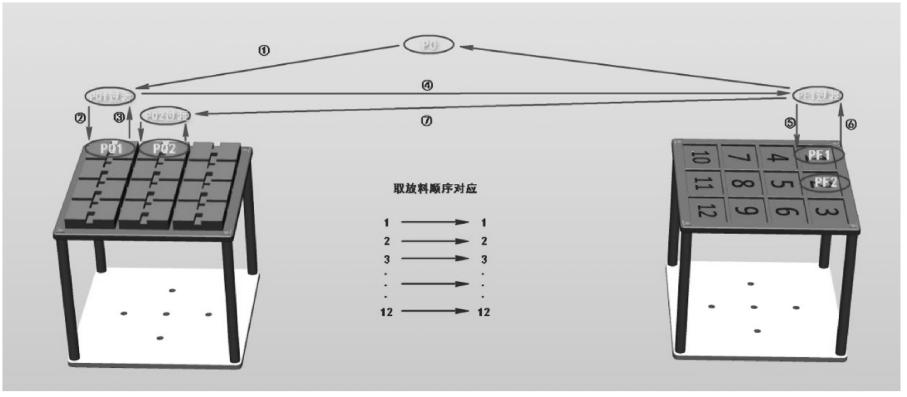


图3-5　物料块搬运整体思路

3.5.1　思路分析

在进行搬运操作时，为了避免工具与工件及其他周边设备发生碰撞，通常需要在合适的位置设置过渡点，工业机器人从起止点(P0)出发，先到达取料点正上方的过渡点(PQ过渡)，然后竖直向下到达取料点(PQ)夹取物料后回到其正上方的过渡点，接着前往放料点正上方的过渡点(PF过渡)，再竖直向下到达放料点并释放物料块后回到其过渡点，最后回到起止点，具体搬运过程如图3-6所示。图片中椭圆中的字符表示点位，圆圈中的数字表示机器人运动的顺序，箭头表示机器人运动的方向。其中过渡点与起止点、过渡点与过渡点之间可以使用MoveJ指令及较高的运行速率；而过渡点与取/放料点之间则需要使用MoveL指令，且速率不能过高。同时，过渡点可以设置合适的转弯半径以提高工作效率。此外，夹具夹取和释放动作需要基于I/O指令实现，且为确保操作可靠，在执行夹取与释放操作前需要设置短暂的延时等待，延时可以通过程序等待指令实现。

根据上述分析，搬运一个物料块需要设置5个目标点，分别是起止点、取料点、放料点、取料点过渡点与放料点过渡点。

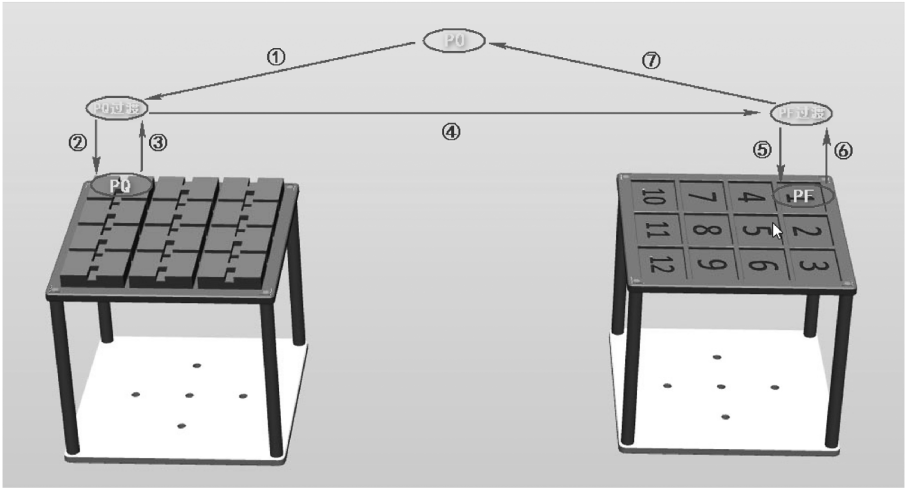


图 3-6　搬运一个物料块示意图

3.5.2　I/O 指令

扫码观看视频
学习I/O指令与程序等待指令

I/O 指令用于控制 I/O 信号，从而实现机器人与工具及周边设备通信，如控制夹具、吸盘等。常用 I/O 指令包括 Set、Reset、WaitDI、WaitDO 等。

(1) Set——置位数字输出信号

◆ 功能：将数字输出信号(digital output)的值设置为 1。

◆ 格式：Set Signal

◆ 参数：

• Signal：有待设置为 1 的信号的名称。【数据类型：signaldo】

(2) Reset——复位数字输出信号

◆ 功能：将数字输出信号的值设置为 0。

◆ 格式：Reset Signal

◆ 参数：

• Signal：有待设置为 0 的信号的名称。【数据类型：signaldo】

◆ 范例

```
Set do1; ! 将数字输出信号 do1 置位为 1
Reset do1; ! 将数字输出信号 do1 复位为 0
```

(3) WaitDI(wait digital input)——等待数字输入信号

◆ 功能：等待数字输入信号输入期望值。

◆ 格式：WaitDI Signal, Value [\MaxTime] [\TimeFlag] [\Visualize] [\Header] [\Message] | [\MsgArray] [\Wrap] [\Icon] [\Image] [\VisualizeTime] [\UIActiveSignal]

◆ 主要参数：

- Signal：信号的名称。【数据类型：signaldi】
- Value：信号的期望值。【数据类型：dionum】
- 其他可选参数可参考指令手册。

(4) WaitDO(wait digital output)——等待数字输出信号

◆ 功能：等待数字输出信号输出期望值。

◆ 格式：WaitDO Signal, Value[\MaxTime][\TimeFlag][\Visualize][\Header][\Message] | [\MsgArray][\Wrap][\Icon][\Image][\VisualizeTime][\UIActiveSignal]

◆ 主要参数：

- Signal：信号的名称。【数据类型：signaldo】
- Value：信号的期望值。【数据类型：dionum】
- 其他可选参数可参考指令手册。

◆ 范例

```
WaitDI di1, 1;
! 等待 di1 的值为 1。如果 di1 为 1，则程序继续往下执行；如果达到最大等待时间(300 s)以后，di1 的值还不为 1，则机器人报警或进入出错处理程序。
WaitDo do1, 1; ! 等待 do1 的值为 1。
```

3.5.3 程序等待指令

工业机器人在进行夹取、释放等操作时，为确保可靠夹取或完全释放，实现对工业机器人作业的精确控制，通常需要用到程序等待指令。RAPID 程序等待的方式较多，除了前述的 I/O 读写等待指令可以基于 I/O 信号来控制程序执行外，还可以通过定时、逻辑状态等程序等待指令来控制程序的执行。主要程序等待指令及其功能如表 3-18 所示。其中，WaitTime、WaitUntil 指令较为常用。

表 3-18 主要程序等待指令及其功能

指令	功能	说明
WaitTime	定时等待	等待给定的时间
WaitUntil	逻辑状态等待	等待直至满足逻辑条件
WaitLoad	程序加载等待	将加载的模块与任务相连
WaitRob	移动到位等待	等待直至达到停止点或零速度
WaitSyncTask	程序同步等待	在同步点等待其他程序任务
WaitSensor	同步监控等待	等待传感器连接
WaitTestAndSet	永久数据等待	等待指定 bool 永久变量值成为 FALSE
WaitWObj	工件等待	等待传送带上的工件

(1) WaitTime

◆ 功能：等待给定的时间。

◆ 格式：WaitTime [\InPos] Time

◆ 参数：

• [\InPos](in position)：如果使用该参数，则机械臂和外轴必须在继续执行之前达到停止点。【数据类型：switch】

• Time：程序执行等待的时间。最短时间(以秒计)为 0，最长时间不受限制，分辨率为 0.001 s。【数据类型：num】

◆ 范例

```
WaitTime 0.5;
! 程序执行等待 0.5 秒。
WaitTime \InPos, 0;
! 程序执行进入等待，直至机械臂和外轴已静止。
```

(2) WaitUntil

◆ 功能：等待直至满足逻辑条件。

◆ 格式：WaitUntil [\InPos] Cond [\MaxTime] [\TimeFlag] [\PollRate] [\Visualize] [\Header] [\Message] | [\MsgArray] [\Wrap] [\Icon] [\Image] [\VisualizeTime] [\UIActiveSignal]

◆ 参数：

• [\InPos](in Position)：如果使用该参数，则机械臂和外轴必须在继续执行之前达到停止点(当前移动指令的 ToPoint)。【数据类型：switch】

• Cond：将等待的逻辑表达式。【数据类型：bool】

• [\MaxTime]：允许的最长等待时间，以秒计。如果在设置条件之前耗尽该时间，则将调用错误处理器，如果不存在错误处理器，则将停止执行。【数据类型：num】

◆ 范例

```
WaitUntil di1=1; ! 当 di4 输入 1 后，继续程序执行。
WaitUntil \Inpos, di2 = 1; ! 程序执行进入等待，直至机械臂已静止，且 di2 输入 1。
WaitUntil di3 = 1 \MaxTime:=5; ! 程序执行进入等待，直至 di3 输入 1。如果已禁用 I/O 设备，或等待时间已到 5 s，则通过错误处理器继续执行。
```

3.5.4　工件坐标与点位创建

在创建目标点位和编程前，要定义三个必要的关键程序数据：工具数据、负荷数据与工件坐标。工具数据在创建工具时已经定义，本任务完成

扫码观看夹具安装与工件坐标创建操作视频

工作坐标创建，并基于工作坐标创建目标点位。

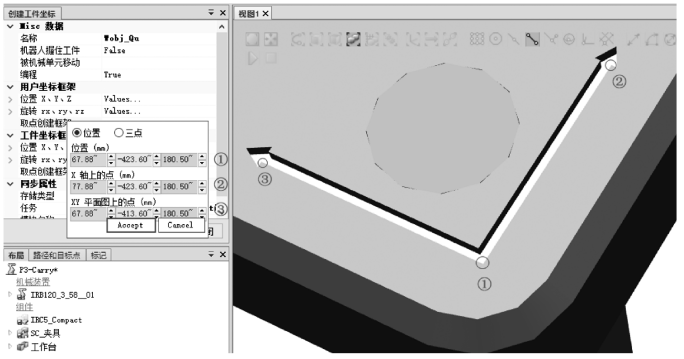
(1)工作坐标创建

工件坐标是用户自定义坐标，表示工件相对大地坐标系的位置。工业机器人可以拥有多个工件坐标，表示不同的工件，或表示同一工件的不同位置。工业机器人编程时就是在工件坐标系中创建目标点和路径。当路径创建完成后如果工件进行了整体移动，只需重新标定工作坐标即可完成轨迹的调整，而无须重新示教目标点位与路径。

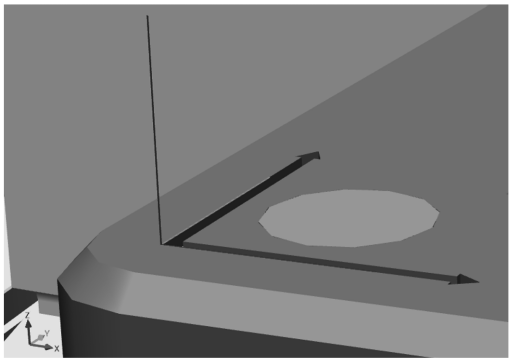
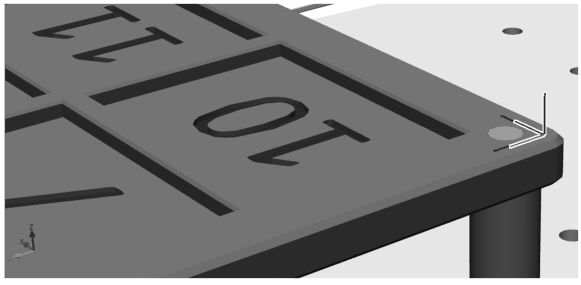
在创建工件坐标时，为了保证定位精度并方便创建或重新标定工件坐标，通常将工件坐标设置于工件表面两相邻棱边垂直的位置，或创建于工件表面专用的定位坐标处，且坐标方向尽量保持与大地坐标方向相同。本项目的物料盘模型上设置有专用的参考坐标，因此可以基于该坐标创建工作坐标。

工件坐标创建有两种方法：位置法与三点法。此处采用位置法，三点法大同小异。由于两个物料盘独立，因此需要在两个物料盘处分别创建工件坐标，创建过程如表 3-19 所示。

表 3-19　创建工作坐标操作步骤

图片示例	步骤说明
	步骤 1：创建工件坐标 1 说明：启动取料盘工件坐标创建并设置名称。 操作：依次选择“基本→其它→创建工件坐标”
	步骤 2 取点创建坐标 说明：在工件台 4 号工位物料盘取点创建工件坐标，工作坐标基于 1 号物料块位置旁的参考坐标。 操作：在“创建工件坐标”面板中，将“Mise 数据”下“名称”的值修改为“Wobj_Qu”，然后选择“用户坐标框架”下的“取点创建框架”，并点击右侧的下拉箭头，选择“位置”，取点如左图所示，确认无误后依次点击“Accept→创建”

续表3-19

图片示例	步骤说明
	步骤 3：确认坐标系 说明：工件坐标创建完成后将视图局部放大，确认坐标系原点与坐标方向是否正确，其中红色坐标轴为 *X* 轴，绿色坐标轴为 *Y* 轴，蓝色坐标轴为 *Z* 轴。
	步骤 4 创建工件坐标 2 说明：以相同的方式，在工件台 2 号工位物料盘上创建另一个工件坐标，命名为“Wobj_Fang”，该工件坐标可以基于 10 号物料位旁的参考坐标创建

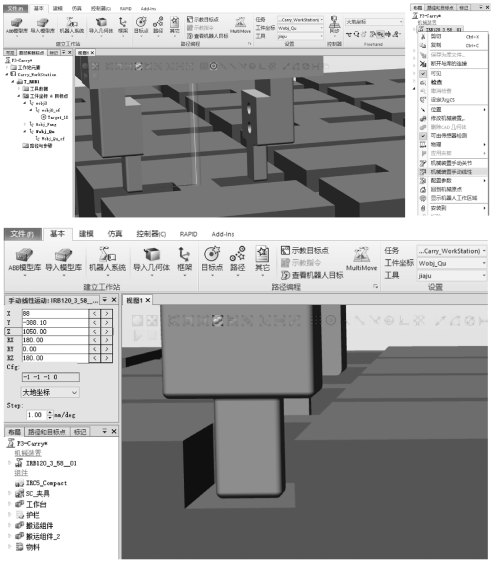
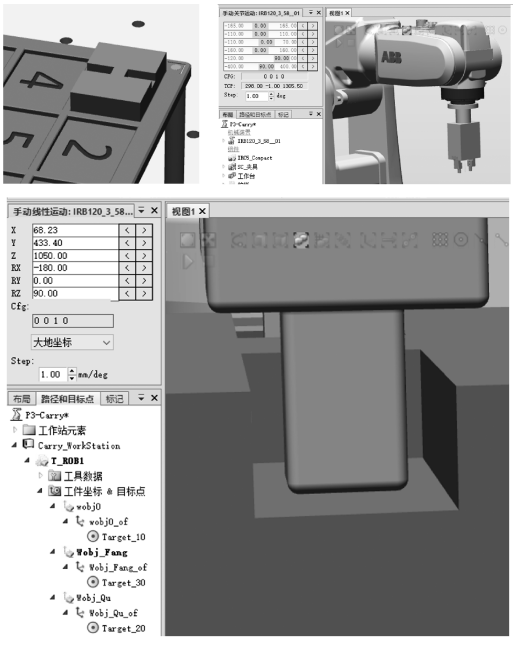
(2)点位示教

本任务中需要创建 5 个目标点，分别是起止点、取料点、放料点、取料点过渡点与放料点过渡点，其中起止点基于工作坐标“wobj0”，取料点及其过渡点基于工件坐标“Wobj_Qu”，放料点及其过渡点基于工件坐标“Wobj_Fang”，点位示教操作步骤如表 3-20 所示。

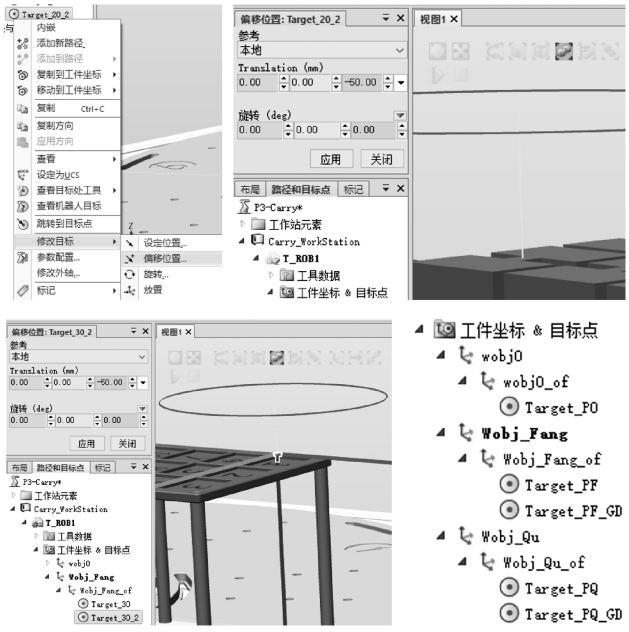
表 3-20　点位示教操作步骤

图片示例	步骤说明
	步骤 1 示教起止点 说明：将机器人初始位置设置为运动起止点。 操作：在“基本”选项卡中，将工件坐标设置为“wobj0”，工具设置为“jiaju”，然后单击“示教目标点”，在弹出的对话框中选择“是”，得到“Target_10”。 注意：如果示教点位前，机器人 TCP 不处于原点位置，可以先让其回到机械原点

续表3-20

图片示例	步骤说明
	步骤 2：调整机器人姿态 说明：调整机器人姿态使夹具竖直向下。 操作：在“布局”面板中右键单击机器人，选择“机械装置手动关节”，在弹出的“手动关节运动”面板中调整第 5 轴(J5，从上至下依次为 J1~J6)至 90°
	步骤 3：示教取料点 说明：在工作台 4 号工位物料盘上 1 号物料块的位置示教取料点，在其正上方示教取料点过渡点。 操作：在“基本”选项卡中，将工件坐标设置为“Wobj_Qu”，工具设置为“jiaju”，然后激活“捕捉中心”，用“手动线性”方式将机器人 TCP 捕捉至 1 号物料块的上表面中心，然后在布局面板中右键单击机器人并选择“机械装置手动线性”，将 TCP 的“Z”值调小 5 mm，确认夹具的位置无误后单击“示教目标点”，得到“Target_20”
	步骤 4 示教放料点 说明：复制物料块并放置于放料盘上的 1 号物料块位置，然后基于该物料块示教放料点。 操作：复制物料块，用两点法将物料块放置到放料盘 1 号物料块位置；让机器人回到“机械原点”并利用“机械装置手动关节”将机器人第 5、6 轴调整为 90°；然后用“手动线性”方式并激活“捕捉中心”将 TCP 捕捉至物料块的上表面中心，再利用“机械装置手动线性”方式沿 Z 轴向下移动 5 mm，将工件坐标设置为“Wobj_Fang”，工具设置为“jiaju”，确认无误后单击“示教目标点”，得到“Target_30”，最后将复制的物料块删除或隐藏

续表3-20

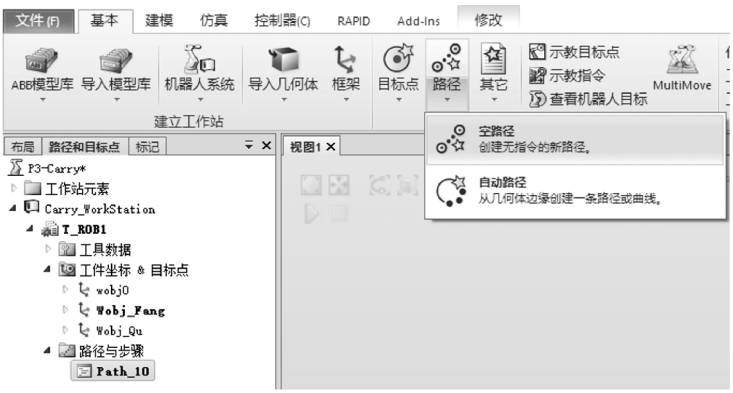
图片示例	步骤说明
	步骤 5 添加过渡点 说明：添加取/放料点的过渡点。 操作：在左侧操作面板中切换至“路径和目标点”，然后依次展开“Carry_WorkStation→T_ROB1→工件坐标 & 目标点”，最后展开各工件坐标即可看到已经示教的目标点。选中“Target_20”并用快捷键“Ctrl+C”复制，然后选中“Wobj_Qu_of”并使用快捷键“Ctrl+V”粘贴，得到“Target_20_2”，然后选中该目标点并单击右键，选择“修改目标→偏移位置”，在“偏移位置”参数面板中参考设置为“本地”，Translation 的“Z”值(第 3 个框)设置为-50，此时视图中可以预览到点位向上偏移了 50mm，确认无误后单击“应用”。同样的方式创建放料点过渡点“Target_30_2”。最后将目标点重命名，起止点为“Target_P0”、取料点为“Target_PQ”、取料点过渡点为“Target_PQ_GD”、放料点为“Target_PF”、放料点过渡点为“Target_PQ_GD”

3.5.5　路径程序创建

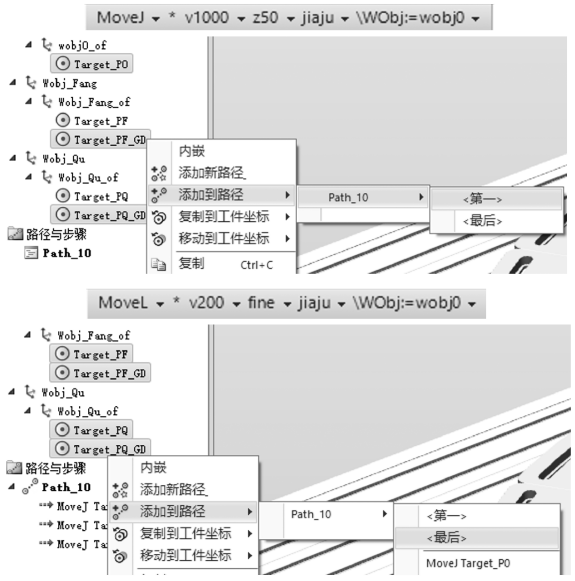
物料块搬运路径程序创建具体步骤如表 3-21 所示。

扫码观看点位示教与路径程序创建操作视频

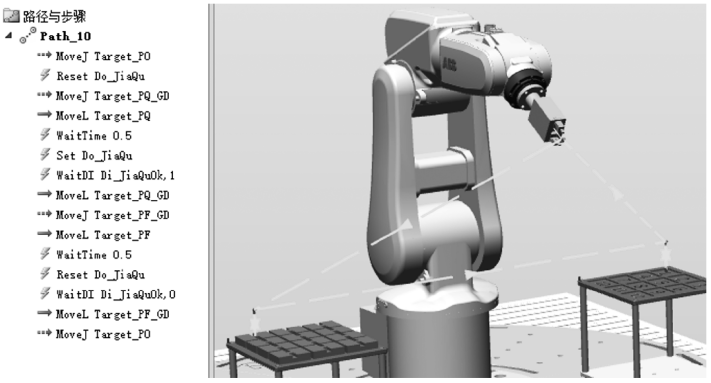
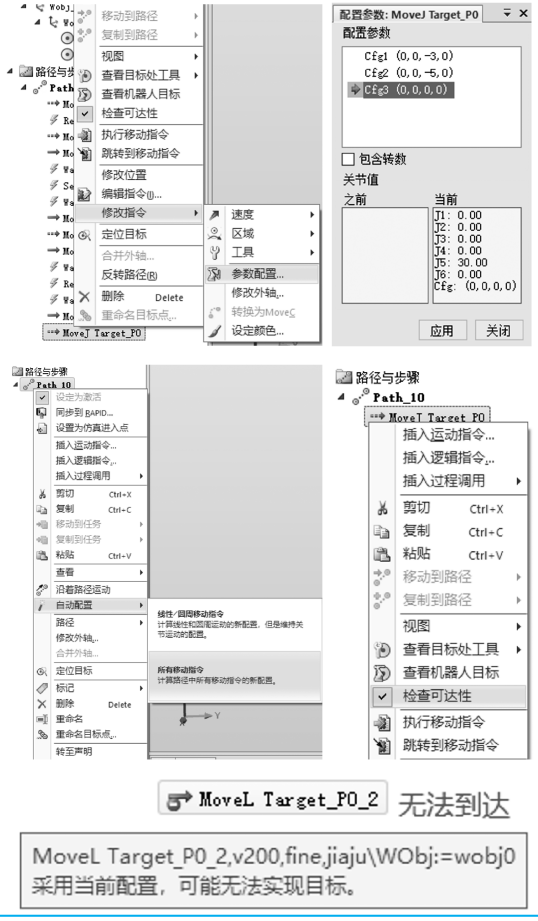
表 3-21　创建 1 个物料块搬运路径程序操作步骤

图片示例	步骤说明
	步骤 1：创建空路径 说明：将机器人初始位置设置为运动起止点。 操作：依次选择“基本→路径→空路径”，此时，会自动创建名称为“Path_10”的空路径

续表3-21

图片示例	步骤说明
	步骤 2：添加运动指令 说明：添加完成 1 个物料块搬运所需的运动指令。 操作：将运动指令参数设置为“MoveJ/v1000/z50”，如左图(1)所示，按下“Ctrl”键同时选中起止点与两个过渡点，单击右键后依次选择“添加到路径→Path_10→第一”；然后将参数设置为“MoveL/v200/fine”，同时选中取料点及其过渡点、放料点及其过渡点，添加到路径“Path_10”的最后
	步骤 3：添加逻辑指令 说明：添加夹具夹取、释放、等待等逻辑指令。 操作：选中“Path_10”并单击右键，选择“插入逻辑指令”，在“创建逻辑指令”面板中“指令模板”选择“Set”，“指令参数”中“Signal”的值选择“Do_JiaQu”，确认无误后单击“创建”。然后继续添加 1 条“Reset”指令，2 条“WaitDI”指令，1 条“WaitTime”指令，参数如左图所示

续表3-21

图片示例	步骤说明
	步骤 4：调整指令顺序 说明：复制部分指令并调整指令顺序。 操作：选中程序指令后按下鼠标左键不放，移动鼠标即可调整指令位置，其中某些重复使用的指令可以先选中，然后用快捷键“Ctrl+C”复制，“Ctrl+V”粘贴，复制运动指令提示是否创建新目标点时选择“否”，最终得到如左图所示的程序。视图中黄色线段即为程序路径指示，其中虚线并不是确定路径，只示意了运动起止点(如 MoveJ 指令)，实线则表示确定路径(如 MoveL 指令)，箭头表示路径方向
	步骤 5：配置指令参数 说明：配置运动指令参数，并确定路径可以正常到达。 操作：选中某条运动指令后单击右键，选择“修改指令→参数配置”，在“配置参数”界面中选择所需的参数并应用，手动配置该条指令的参数。也可选中路径“Path_10”并单击右键，选择“自动配置→所有移动指令”，即可自动配置所有运动指令参数。最后，选中一条运动指令并单击右键，确认“检查可达性”被勾选(默认已勾选)，如果所有运动指令没有出现红色标识(如左边最下方图片)即说明所有目标点均可达到

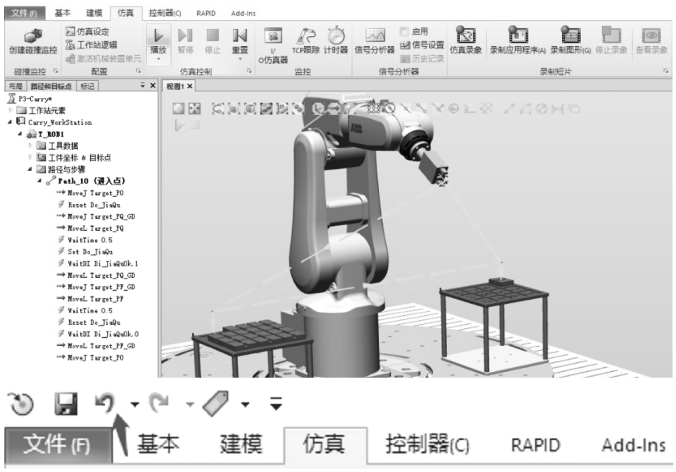
3.5.6 同步 Rapid 与仿真运行

当前程序数据与程序指令都只存在于工件站中，为了实现动态仿真运行，需要将工作站中的数据同步到虚拟控制器中。同步数据后将编制的程序“Path_10”设置为仿真进入点即可进行仿真，具体操作过程如表 3-22 所示。

表 3-22 同步 Rapid 与仿真运行操作步骤

图片示例	步骤说明
	步骤 1：同步到 Rapid 说明：将目标点位与程序指令等同步到虚拟控制器中。 操作：依次选择“基本→同步→同步到 RAPID”(也可以选中路径“Path_10”并单击右键选择“同步到 RAPID”)，在弹出的“同步到 RAPID”对话框中将需要同步的数据选中(首次同步时建议都选中)，单击“确定”
	步骤 2：设置仿真进入点 说明：将路径“Path_10”设置为仿真进入点。 操作：依次选择“仿真→仿真设定→T_ROB1”，将进入点选择为“Path_10”，然后关闭仿真设定(也可以选中路径“Path_10”并单击右键选择“设置为仿真进入点”)，当“Path_10”被标注为“进入点”即表示设置成功

续表3-21

图片示例	步骤说明
	步骤 3：仿真运行 说明：仿真运行并完成 1 个物料块的搬运任务。 操作：依次选择“仿真→播放”，机器人即按照程序规划的轨迹完成搬运物料块 1 的任务。为了方便后面任务的调试，仿真结束后点击“撤销”（或使用快捷键 Ctrl+Z），使布局撤回至搬运前状态

任务 6　搬运 1 行

3.6.1　思路分析

取料盘上 1 行共有 3 个物料块，因此 1 行的搬运其实就是搬运 1 个物料块的操作重复 3 次。为了提高工作效率，在完成第 1 个物料块搬运并回到放料点正上方过渡点后，直接去第 2 个物料块的取料过渡点，完成第 2 个物料块搬运后直接前往搬运第 3 个物料块，取/放物料块的编号一一对应，完成整行的搬运后再回到起止点，搬运思路如图 3-7 所示。

根据上述思路分析，并结合任务 5 中的搬运程序，除去共用的起止点，1 个物料块需要设置 4 个目标点（取、放料点及其各自过渡点），因此完成 1 行的搬运还需要示教 8 个点位，而如果要完成整盘的搬运则还需要创建更多的点位，显然示教点位的工作会很烦琐。在放置物料块时已经知道各物料块位置存在一定规律，行列方向均间隔 5 mm，物料块长 55 mm、宽 40 mm，很容易计算出各物料块夹持位置在列方向上间隔 60 mm，行方向上间隔 45 mm。因此根据其布局规律，可以使用偏移功能与循环指令，基于物料块 1 的位置进行循环偏移完成其他物料块的搬运，而且过渡点也可以基于取料点向上偏移 50 mm。本任务中，假设列值用 m 表示，且首列为 0，则每一列相对于首列的偏移值为 60 m。

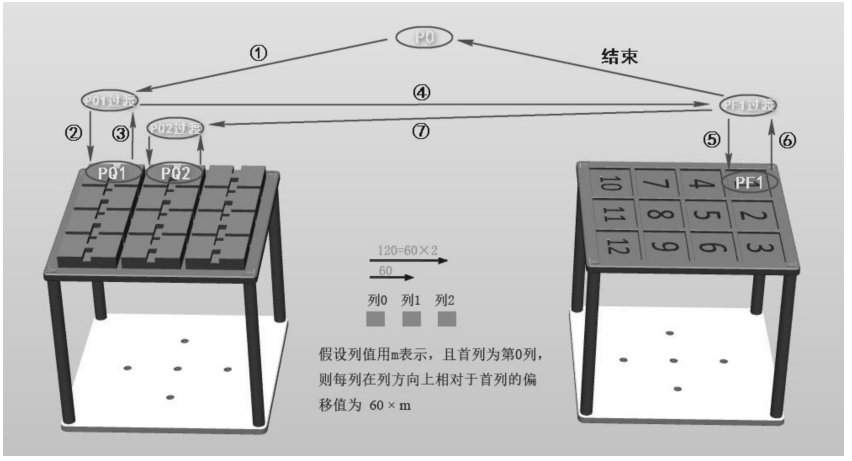


图 3-7 搬运 1 行物料思路示意图

3.6.2 Offs——偏移功能函数

扫码观看视频学习偏移功能函数与逻辑控制指令

◆ 功能：在一个机械臂位置的工件坐标系中添加一个偏移量。

◆ 格式：Offs（Point，XOffset，YOffset，ZOffset）

◆ 参数：

- Point 有待移动的位置数据。【数据类型：robtarget】
- XOffset 工件坐标系中 *X* 方向的位移。【数据类型：num】
- YOffset 工件坐标系中 *Y* 方向的位移。【数据类型：num】
- ZOffset 工件坐标系中 *Z* 方向的位移。【数据类型：num】

◆ 范例

MoveL Offs(P1，3，5，10)，v100，z0，tool1；

！将机械臂移动至 P1 点 *X* 轴正方向 3 mm，*Y* 轴正方向 5 mm，*Z* 轴正方向 10 mm 的一个点。

P3:=Offs(P2，5，-10，15)；

！将位置 P2 沿 *X* 方向移动 5 mm，沿 *Y* 负方向移动 10 mm，且沿 *Z* 方向移动 15 作为 P3 的位置。

3.6.3 逻辑控制指令

(1) if 指令

◆ 功能：根据是否满足条件，执行不同的指令。

◆ 格式：

- 单分支格式

```
IF <EXP> THEN
    <SMT>
ENDIF
```

◆ 范例

```
IF reg1 > 5 THEN
    Set do1;
ENDIF
! 当 reg1 大于 5 时，将信号 do1 置位
IF counter > 100 THEN
    counter:=100;
ELSEIF counter < 0 THEN
    counter:=0;
ELSE
    counter:=counter + 1;
ENDIF
! 当 counter 的值大于 100 时将其赋值为 100，如果小于 0 则赋值为 0，否则将其值加
```

- 双分支格式

```
IF <EXP> THEN
    <SMT>
ELSE
    <SMT>
ENDIF
```

- 多分支格式

```
IF<EXP> THEN
    <SMT>
ELSEIF <EXP> THEN
    <SMT>
ELSE
    <SMT>
ENDIF
```

(2) compact if 指令

◆ 功能：紧凑型 if 指令，当单个指令仅在满足给定条件的情况下执行时使用。

◆ 格式：IF <EXP> <SMT>

◆ 范例

```
IF reg1 > 5 Set do1;
! 当 reg1 大于 5 时，将信号 do1 置位，与下面的程序功能相同。
IF reg1 > 5 THEN
    Set do1;
ENDIF
```

(3) test 指令

◆ 功能：根据表达式或数据的值选择执行相应的指令，如图 3-8 所示。

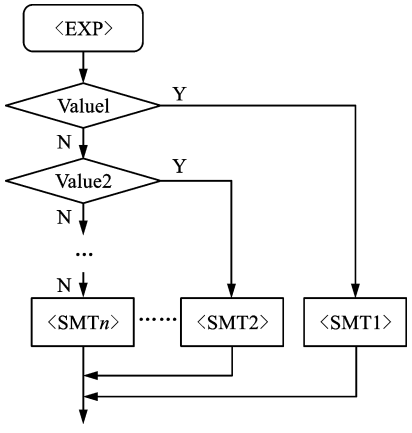


图 3-8　test 指令执行流程图

◆ 格式：

```
TEST <EXP>
  CASE <Value1>:
      <SMT1>
  CASE <Value2>:
      <SMT2>
  ……
  DEFAULT:
      <SMTn>
ENDTEST
```

◆ 范例

```
TEST reg1
    CASE 1:
        routine1;
    CASE 2,3:
        routine2;
    DEFAULT:
        routine3;
ENDTEST
! 如果 reg1 的值为 1，则执行 routine1；如果其值为 2 或 3 时，则执行 routine2；否则执行 routine3。
```

(4)for 指令

◆ 功能：根据循环变量在指定范围内递增或递减而重复执行语句块，主要用于一个或多个指令需要重复执行数次(执行次数确定)的情况。

◆ 格式：

FOR <循环变量> FROM <初始值> TO <终止值> [STEP<步长>] DO
    <语句块>
ENDFOR

◆ 说明：指令中的循环变量会自动声明，无须单独定义，且初始值、终止值与步长值通常都为整数值。循环开始时，循环变量从初始值开始，如果未指定 STEP 步长值，则默认 STEP 值为 1；如果是递减的情况，STEP 值可设为负数。每次循环时，都要重新计算循环变量，只要变量不在循环范围内，循环即结束，继续执行后续语句。

◆ 范例

```
FOR i FROM 1 TO 10 DO
        routine1;
ENDFOR
! 程序 routine1 重复执行 10 次。
```

(5)while 指令

◆ 功能：用于在给定条件满足的情况下重复执行某些指令语句。

◆ 格式：

WHILE <条件表达式> DO
    <语句块>
ENDWHILE

◆ 说明：条件满足(条件表达式为真)时，程序会一直重复执行语句块。一旦条件不满足(表达式求值为假)，循环结束，继续执行后续指令。

◆ 范例

```
WHILE reg1 < reg2 DO
        reg1 := reg1 + 1;
ENDWHILE
! 只要 reg1 < reg2，reg1 的值循环加 1。
```

3.6.4　Rapid 程序与管理

扫码观看视频学习Rapid程序与管理

当程序功不断完善，程序代码也会越来越长，控制逻辑也会更复杂，在工作站中以插入指令的形式编写复杂程序会相对困难，同时修改和调试程序也会相对麻烦。通常编写复杂程序时会直接在 Rapid 程序编辑器中进行，进入 Rapid 程序编辑器的操作如图 3-9 所示，依次选择“RAPID→RAPID→T_ROB1→Module1”，双击“main”或“Path_10”即可进入 Rapid 程序编辑器界面。

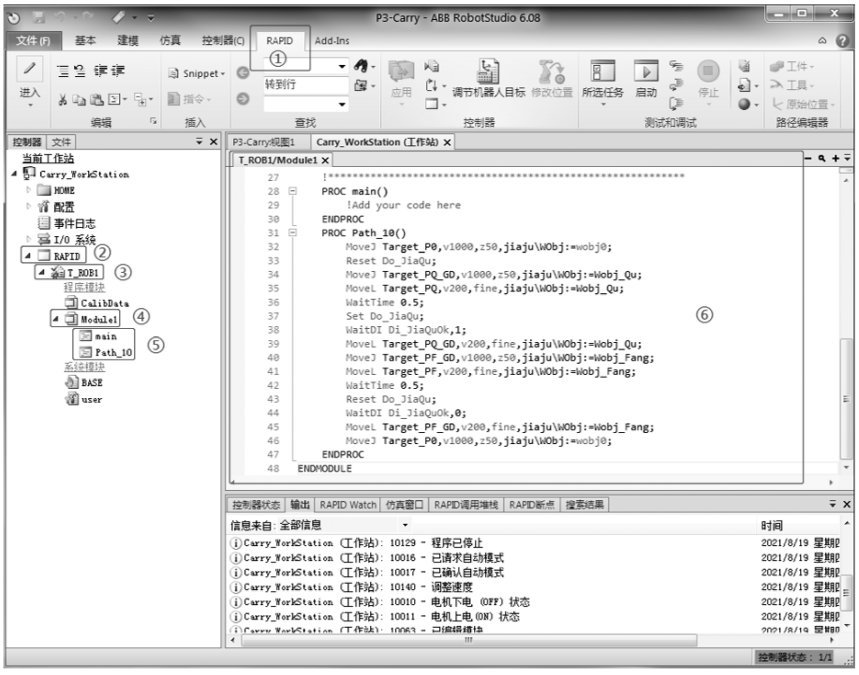


图 3-9　Rapid 程序编辑器界面

RAPID 语言是 ABB 工业机器人平台的编程语言，利用 RAPID 语言编制的程序简称 RAPID 程序，一个 RAPID 程序称为一个任务，一个任务是由一系列模块组成，包括程序模块与系统模块，其组成如图 3-10 所示。一般地，我们只通过新建程序模块来构建机器人程序，而系统模块多用于系统方面的控制。

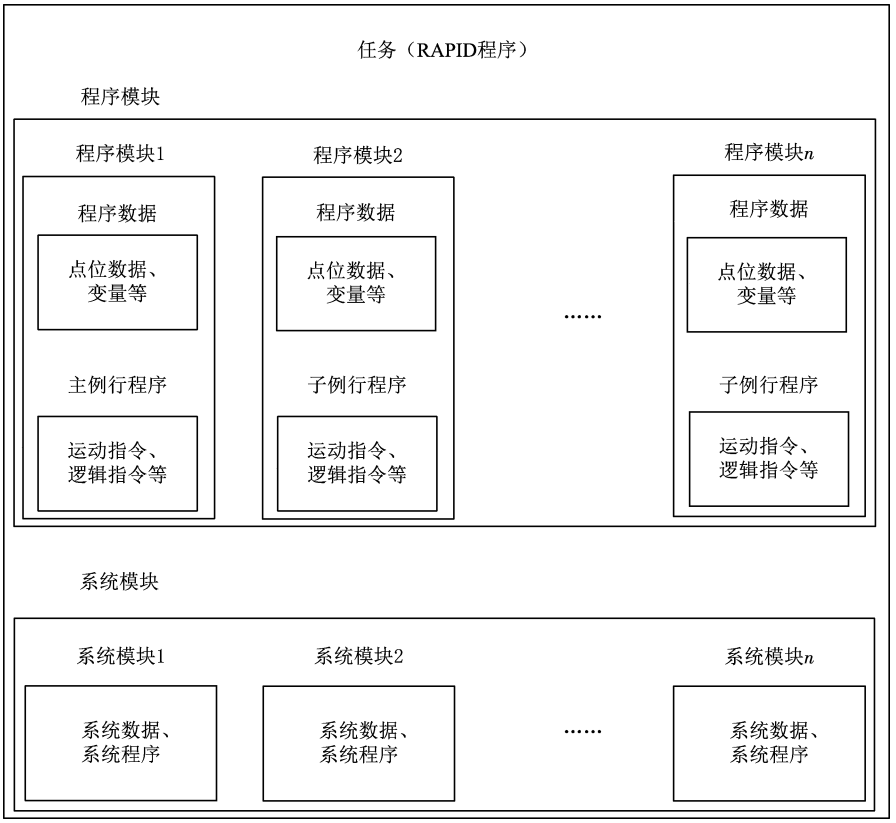


图 3-10　RAPID 程序组成示意图

在实际应用中可以根据不同的用途创建多个程序模块，如专门用于主控制的程序模块，用于存放数据的程序模块和用于位置计算的程序模块，这样可以方便归类管理不同用途的例行程序与数据。每个程序模块由程序数据和程序两部分组成。

1. 程序

根据功能与用途的不同，程序又分为普通程序(例行程序)PROC、中断程序 TRAP 和功能程序 FUNC，一个模块中并不一定同时包含这 3 种程序。程序模块之间的数据、例行程序、中断程序和功能程序可以互相调用。

通常例行程序又分为主例行程序与子例行程序。在 RAPID 程序中，只能有一个主例行程序 main，可以存在于任意一个程序模块中，主例行程序作为整个 RAPID 程序执行的起点。子例行程序则可以有多个，用于实现特定的控制功能，通常被主例行程序调用，从而实现模块化编程。

(1)例行程序(PROC)

RAPID 主程序以及大多数子程序均为例行程序(PROC)，例行程序可以被其他模块或程序调用，但不能向主调程序返回数据，故又称为无返回值程序。例行程序以 PROC 开始，ENDPROC 结束，可以定义参数，不使用

参数时也需保留“( )”，其结构与格式如下：

PROC 程序名称(参数表)

　　程序指令

　　……

ENDPROC

(2)中断程序(TRAP)

中断程序通常用于事件(信号)的优先处理，ABB 工业机器人程序是逐行执行的，不同于 PLC(实时扫描)，所以在遇到一些需要即时响应的状态时就要使用中断程序来触发。触发中断时，机器人会停止目前执行的程序，转而执行中断程序，中断程序执行完毕后回到原程序继续执行，其执行流程如图 3-11 所示。

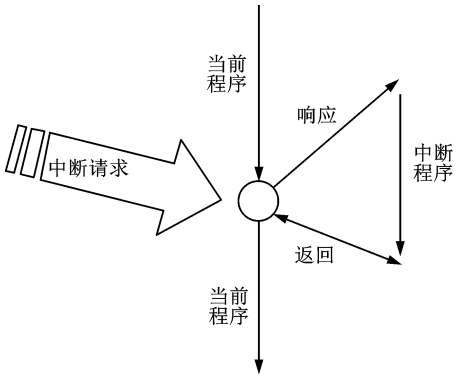


图 3-11　中断执行流程

(3)功能程序(FUNC)

功能程序(FUNC)又称有返回值程序，这是一种具有运算、比较等功能，能向调用该程序的模块、程序返回结果的参数化编程模块；调用功能程序时，不仅需要指定程序名称，且必须有程序参数。

全局功能程序直接以程序类型 FUNC 开始，用 ENDFUNC 结束，程序结构与格式如下：

FUNC 数据类型 功能名称

　　程序数据定义

　　程序指令

　　……

　　RETURN 返回数据

ENDFUNC

2. 程序数据

程序数据是在程序模块或系统模块中设定的值和定义的一些环境数据，创建的程序数据由同一个模块或其他模块中的指令调用。了解程序数据是 ABB 机器人编程的基础。

在声明程序数据时，必须设置其所属任务、模块、范围、数据类型与存储类型等属性，其中需要重点了解程序数据类型与存储类型。

(1)程序数据类型

ABB 机器人程序数据类型有 100 余种，可以分为基本数据、I/O 数据、运动相关数据等几个大类，其中常用的数据类型如表 3-23 所示。

表 3-23　常用数据类型

大类	数据类型	说明
基本数据	bool	逻辑值。False 或 0 表示逻辑假，True 或 1 表示逻辑真
	byte	字节值。表示一个整数字节值，取值范围为(0~255)
	num	数值。存储整数、小数等，取值范围(-8388607 ~ +8388608)
	String	字符串。可由字母、数字、符号等组成，必须包含于双引号中，最多 80 个字符
I/O 数据	dionum	数字值。用于数字 I/O 信号，取值为 0 或 1
	signaldi/do	数字量输入/输出信号。取值为 0 或 1，低电平为 0，高电平为 1
	signalgi/go	数字量输入/输出信号组。多个数字输入或输出信号组合使用
运动相关数据	robtarget	位置数据。用于定义机械臂和附加轴的位置，通常包括 TCP 基于工件坐标的空间位置(*X*/*Y*/*Z*)、工具方位、机械臂的轴配置等数据
	jointtarget	接头位置数据(关节位置数据)。机械臂轴的轴位置以度数计。轴位置定义为各轴(臂)从轴校准位置沿正方向或反方向旋转的度数
	tooldata	工具数据。用于描述工具的特征，包括 TCP 的位置、方位以及工具负载的物理特征
	wobjdata	工件数据。用于定义工件的位置及状态
	zonedata	区域数据。用于规定如何结束一个位置，即在朝下一个位置移动前，轴必须如何接近编程位置。系统中定义了一系列区域数据，如停止点 fine 或飞越点(z0、z1、z10 等)
	loaddata	负载数据。描述附于机械臂机械界面(机械臂安装法兰)的负载，同时将 loaddata 作为 tooldata 的组成部分，以描述工具负载

(2)存储类型

在定义程序数据时，除了要确定数据类型，还要指定其存储类型。ABB机器人数据存储类型有3种，分别如表3-24所示。

表3-24　存储类型

存储类型	名称	说明
VAR	变量	在程序执行的过程中和停止时，会保持当前的值。但在程序指针复位后，将恢复为初始值
PERS	可变量	无论程序的指针如何改变位置，可变量型程序数据都会保持其最后赋予的值
CONST	常量	数据在定义的时候已经被赋予了数值，且不能在程序中进行数据的修改，除非手动进行修改，否则数据会一直保持不变

3. 运算符与表达式

在RAPID程序中，程序数据的值可以直接通过赋值指令获得，也可以利用表达式、运算指令或函数命令进行数学、逻辑运算得到。简单四则运算和比较操作可使用基本运算符，复杂运算则需要使用函数命令。基本运算符如表3-25所示。

表3-25　基本运算符

运算符		运算	运算数类型	举例
算术运算符	:=	赋值	任意	a:=2; b:=c
	+	加	num、dnum、pos、String	1+2; a+3; a+b
	-	减	num、dnum、pos	5-2; 6-a; a-b
	*	乘	num、dnum、pos、orient	a:=2*3;
	/	除	num、dnum	a:=6/2
比较运算符	<	小于	num、dnum	a<b; (1<2)=True
	<=	小于等于	num、dnum	a<=b; (2<=1)=False
	>	大于	num、dnum	a>b; (1>2)=False
	>=	大于等于	num、dnum	a>=b; (1>=2)=False
	=	等于	任意同类型	(1=2)=False; a=b
	<>	不等于	任意同类型	a<>b; (1<>2)=True

扫码观看搬运1行的分析与操作视频

3.6.5 编程与调试

(1)基于 FOR 搬运 1 行的程序

```
PROC Path_10( )
    Reset Do_JiaQu;
    MoveJ Target_P0,v1000,z50,jiaju\WObj:=wobj0;
    FOR m FROM 0 TO 2 DO
      MoveJ offs(Target_PQ,0,60*m,50),v1000,z50,jiaju\WObj:=Wobj_Qu;
      MoveL offs(Target_PQ,0,60*m,0),v200,fine,jiaju\WObj:=Wobj_Qu;
          WaitTime 0.5;
          Set Do_JiaQu;
          WaitDI Di_JiaQuOk,1;
          MoveL offs(Target_PQ,0,60*m,50),v200,fine,jiaju\WObj:=Wobj_Qu;
          MoveJ offs(Target_PF,60*m,0,50),v1000,z50,jiaju\WObj:=Wobj_Fang;
          MoveL offs(Target_PF,60*m,0,0),v200,fine,jiaju\WObj:=Wobj_Fang;
          WaitTime 0.5;
          Reset Do_JiaQu;
          WaitDI Di_JiaQuOk,0;
          MoveL offs(Target_PF,60*m,0,50),v200,fine,jiaju\WObj:=Wobj_Fang;
      ENDFOR
      MoveJ Target_P0,v1000,z50,jiaju\WObj:=wobj0;
ENDPROC
```

(2)基于 While 循环搬运 1 行的程序

```
PROC Path_10( )
    VAR num num1:=0;
    Reset Do_JiaQu;
    MoveJ Target_P0,v1000,z50,jiaju\WObj:=wobj0;
    while num1<=2 DO
        MoveJ offs(Target_PQ,0,60*num1,50),v1000,z50,jiaju\WObj:=Wobj_Qu;
        MoveL offs(Target_PQ,0,60*num1,0),v200,fine,jiaju\WObj:=Wobj_Qu;
        WaitTime 0.5;
        Set Do_JiaQu;
        WaitDI Di_JiaQuOk,1;
        MoveL offs(Target_PQ,0,60*num1,50),v200,fine,jiaju\WObj:=Wobj_Qu;
        MoveJ offs(Target_PF,60*num1,0,50),v1000,z50,jiaju\WObj:=Wobj_Fang;
        MoveL offs(Target_PF,60*num1,0,0),v200,fine,jiaju\WObj:=Wobj_Fang;
        WaitTime 0.5;
        Reset Do_JiaQu;
        WaitDI Di_JiaQuOk,0;
        MoveL offs(Target_PF,60*num1,0,50),v200,fine,jiaju\WObj:=Wobj_Fang;
        num1:=num1+1;
    ENDWHILE
    MoveJ Target_P0,v1000,z50,jiaju\WObj:=wobj0;
ENDPROC
```

任务7　搬运整盘

扫码观看搬运整盘的分析与操作视频

3.7.1　思路分析

搬运整盘其实就是搬运1行并重复4次，已知每行间隔45 mm，因此每完成1行的搬运后在行方向上进行45 mm偏移，即可继续下一行搬运，假设行值用 n 表示，且首行为第0行，则每行在行方向上相对于首行的偏移值为45 n，如图3-12所示。前一任务中已经实现了一层循环控制搬运程序在列方向上进行循环偏移，整盘搬运时还需要在行方向上进行偏移，因此需要两层循环，即循环嵌套。

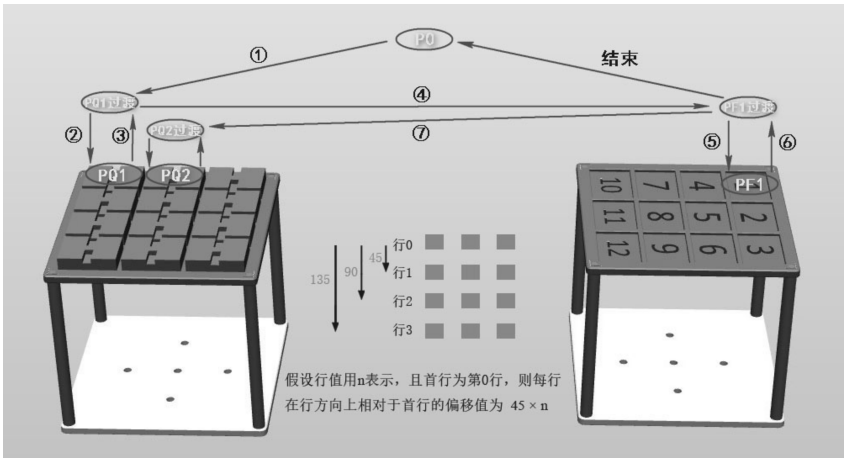


图3-12　整盘搬运思路示意图

3.7.2　循环嵌套

循环嵌套，即在一个循环体语句中又包含另一个循环语句。循环嵌套可以是两层循环，也可以是多层循环，可以是FOR循环与FOR循环嵌套、While循环与While循环嵌套，也可以是FOR循环与While循环嵌套。下面以两层FOR循环嵌套为例说明循环嵌套的使用与程序执行流程。

(1)循环嵌套格式

```
FOR x FROM a TO b DO
   FOR y FROM c TO d DO
      循环体语句;
   ENDFOR
ENDFOR
```

(2)控制流程

假定循环嵌套中a<b，c<d，则其执行流程如图3-13所示。

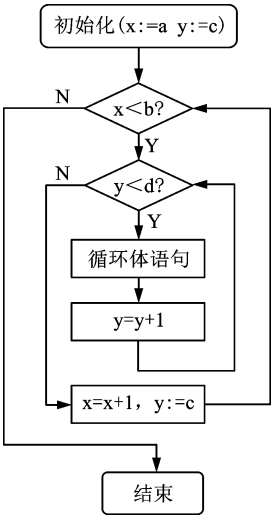


图 3-13 循环嵌套控制流程图

3.7.3 编程与调试

以下是使用双层 FOR 循环嵌套实现整盘搬运的程序，可以自行尝试双层 While 循环嵌套，以及 FOR 循环与 While 循环嵌套实现整盘搬运的程序编写与调试。

```
PROC Path_40( )
  IF Do_JiaQu=1 THEN
    Reset Do_JiaQu;
  ENDIF
  MoveJ Target_P0,v1000,z50,jiaju\WObj:=wobj0;
  FOR n FROM 0 TO 3 DO
    FOR m FROM 0 TO 2 DO
      MoveJ offs(Target_PQ,45*n,60*m,50),v1000,z50,jiaju\WObj:=Wobj_Qu;
      MoveL offs(Target_PQ,45*n,60*m,0),v200,fine,jiaju\WObj:=Wobj_Qu;
      WaitTime 0.5;
      Set Do_JiaQu;
      WaitDI Di_JiaQuOk,1;
      MoveL offs(Target_PQ,45*n,60*m,50),v200,fine,jiaju\WObj:=Wobj_Qu;
      MoveJ offs(Target_PF,60*m,-45*n,50),v1000,z50,jiaju\WObj:=Wobj_Fang;
      MoveL offs(Target_PF,60*m,-45*n,0),v200,fine,jiaju\WObj:=Wobj_Fang;
      WaitTime 0.5;
      Reset Do_JiaQu;
      WaitDI Di_JiaQuOk,0;
      MoveL offs(Target_PF,60*m,-45*n,50),v200,fine,jiaju\WObj:=Wobj_Fang;
    ENDFOR
  ENDFOR
  MoveJ Target_P0,v1000,z50,jiaju\WObj:=wobj0;
ENDPROC
```

任务8　往返搬运

扫码观看往返搬运的分析与操作视频

3.8.1　思路分析

为了仿真的方便，可以在完成整盘物料的搬运后再将其搬回，方便下一次的搬运仿真。往返搬运主要包括正常的搬运与搬回两步操作，而搬回其实就是正常搬运的逆操作，只需将点位与偏移参数进行互换即可。为了使程序结构更清晰易读，可以使用子程序。

3.8.2　子程序应用

在模块化编程中，通常将某些功能程序，尤其是需要多次重复执行的程序编制为子程序，然后在主程序中调用各子程序，提高程序代码的复用率，使程序结构更加清晰，且方便阅读与修改。本任务中将初始化程序、搬运程序、搬回程序分别定义为子程序，然后在主程序中调用完成往返搬运。此外还可以利用传参数的方式将搬运和搬回两个子程序整合为一个子程序，供学习者参考。

3.8.3　编程与调试

(1)无参程序

```
MODULE Module1
  ! 此处省略了从工作站同步到虚拟控制器的 Target_P0、Target_PQ、Target_PF 等目标点位数据
  ! 初始化子程序
  PROC init( )
    Reset Do_JiaQu;
    MoveJ Target_P0, v1000, z50, jiaju\WObj: =wobj0;
  ENDPROC
  ! 搬运子程序
  PROC carryTo( )
    FOR n FROM 0 TO 3 DO
      FOR m FROM 0 TO 2 DO
        MoveJ offs(Target_PQ, 45 * n, 60 * m, 50), v1000, z50, jiaju\WObj: =Wobj_Qu;
        MoveL offs(Target_PQ, 45 * n, 60 * m, 0), v200, fine, jiaju\WObj: =Wobj_Qu;
        WaitTime 0.5;
        Set Do_JiaQu;
        WaitDI Di_JiaQuOk, 1;
        MoveL offs(Target_PQ, 45 * n, 60 * m, 50), v200, fine, jiaju\WObj: =Wobj_Qu;
        MoveJ offs(Target_PF, 60 * m, -45 * n, 50), v1000, z50, jiaju\WObj: =Wobj_Fang;
        MoveL offs(Target_PF, 60 * m, -45 * n, 0), v200, fine, jiaju\WObj: =Wobj_Fang;
        WaitTime 0.5;
```

```
            Reset Do_JiaQu;
            WaitDI Di_JiaQuOk, 0;
            MoveL offs(Target_PF, 60 * m, -45 * n, 50), v200, fine, jiaju\WObj:=Wobj
_Fang;
        ENDFOR
      ENDFOR
      MoveJ Target_P0, v1000, z50, jiaju\WObj:=wobj0;
    ENDPROC
    ! 搬回子程序
    PROC carryBack()
      FOR n FROM 0 TO 3 DO
        FOR m FROM 0 TO 2 DO
          MoveJ offs(Target_PF, 60 * m, -45 * n, 50), v1000, z50, jiaju\WObj:=
Wobj_Fang;
          MoveL offs(Target_PF, 60 * m, -45 * n, 0), v200, fine, jiaju\WObj:=Wobj_
Fang;
          WaitTime 0.5;
          set Do_JiaQu;
          WaitDI Di_JiaQuOk, 1;
          MoveL offs(Target_PF, 60 * m, -45 * n, 50), v200, fine, jiaju\WObj:=Wobj
_Fang;
        MoveJ offs(Target_PQ, 45 * n, 60 * m, 50), v1000, z50, jiaju\WObj:=Wobj_Qu;
          MoveL offs(Target_PQ, 45 * n, 60 * m, 0), v200, fine, jiaju\WObj:=Wobj_Qu;
          WaitTime 0.5;
          reSet Do_JiaQu;
          WaitDI Di_JiaQuOk, 0;
          MoveL offs(Target_PQ, 45 * n, 60 * m, 50), v200, fine, jiaju\WObj:=Wobj_Qu;
        ENDFOR
      ENDFOR
      MoveJ Target_P0, v1000, z50, jiaju\WObj:=wobj0;
ENDPROC
! 主程序
PROC main()
      init;    ! 调用初始化子程序
      carryTo;    ! 调用搬运子程序
      carryBack;    ! 调用搬回子程序
    ENDPROC
ENDMODULE
```

(2)有参程序

```
MODULE Module1
    ! 此处省略了从工作站同步到虚拟控制器的 Target_P0、Target_PQ、Target_PF 等目
标点位数据
    PERS wobjdata Wobj_Pick;
    PERS wobjdata Wobj_Release;
    ! 初始化子程序
    PROC init()
```

```
    Reset Do_JiaQu;
    MoveJ Target_P0, v1000, z50, jiaju\WObj:=wobj0;
  ENDPROC
  ! 搬运子程序
  PROC carry(num a)  ! 搬运子程序的参数为 1 时表示正常搬运，为其他值时则表示往回搬运
    VAR robtarget target_Pick;
    VAR robtarget target_Release;
    VAR num symbol1:=0;
    VAR num symbol2:=0;
    VAR num offs1:=0;
    VAR num offs2:=0;
    IF a=1 THEN
      target_Pick:=Target_PQ;
      target_Release:=Target_PF;
      Wobj_Pick:=Wobj_Qu;
      Wobj_Release:=Wobj_Fang;
      symbol1:=1;
      symbol2:=-1;
    ELSE
      target_pick:=Target_PF;
      target_Release:=Target_PQ;
      Wobj_Pick:=Wobj_Fang;
      Wobj_Release:=Wobj_Qu;
      symbol1:=-1;
      symbol2:=1;
    ENDIF
    FOR n FROM 0 TO 3 DO
      FOR m FROM 0 TO 2 DO
        IF a=1 THEN
          offs1:=45*n;
          offs2:=60*m;
        ELSE
          offs1:=60*m;
          offs2:=45*n;
        ENDIF
        MoveJ offs(target_pick, offs1, offs2*symbol1, 50), v1000, z50, jiaju\WObj:=Wobj_Pick;
        MoveL offs(target_pick, offs1, offs2*symbol1, 0), v200, fine, jiaju\WObj:=Wobj_Pick;
        WaitTime 0.5;
        Set Do_JiaQu;
        WaitDI Di_JiaQuOk, 1;
        MoveL offs(target_pick, offs1, offs2*symbol1, 50), v200, fine, jiaju\WObj:=Wobj_Pick;
        MoveJ offs(target_Release, offs2, offs1*symbol2, 50), v1000, z50, jiaju\WObj:=Wobj_Release;
        MoveL offs(target_Release, offs2, offs1*symbol2, 0), v200, fine, jiaju\WObj:=Wobj_Release;
        WaitTime 0.5;
```

```
            Reset Do_JiaQu;
            WaitDI Di_JiaQuOk, 0;
            MoveL offs(target_Release, offs2, offs1 * symbol2, 50), v200, fine, jiaju\
WObj:=Wobj_Release;
        ENDFOR
    ENDFOR
    MoveJ Target_P0, v1000, z50, jiaju\WObj:=wobj0;
  ENDPROC
  ! 主程序
  PROC main()
    init;   ! 调用初始化子程序
    carry(1);   ! 调用搬运子程序完成初始搬运
    carry(2);   ! 调用搬运子程序将物料搬回
  ENDPROC
ENDMODULE
```

项目总结

本项目以工业机器人搬运工作站为任务载体，主要包括搬运工作站布局、动态夹具创建、工作站逻辑设定、搬运程序编写与调试等工作任务，其中在项目实施过程中，将动态夹具创建细分为夹具工具创建与夹具夹取属性创建两个任务，搬运程序编写与调试则细分为搬运 1 个、搬运 1 行、搬运整盘、往返搬运四个任务。

项目涉及的主要理论知识都融入到了各个任务中，本项目中需要重点掌握 Smart 组件、I/O 指令、程序等待指令、逻辑控制指令、偏移功能函数(Offs)的功能与使用方法。

任务拓展

扫码观看视频了解区域数据(zonedata)

◆ 区域数据-zonedata

(1)区域数据(zonedata)概述

区域数据(zonedata)用于规定如何结束一个位置，即在朝下一个位置移动之前，轴必须如何接近当前位置。ABB 工业机器人可以用停止点或飞越点的形式来终止一个位置，如图 3-14 所示。

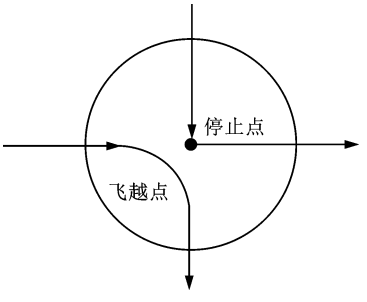


图 3-14　停止点与飞越点效果示意图

停止点：指在继续执行下一个程序指令前，机器人和附加轴必须完全到达指定位置并短暂停止。

飞越点：意味着机器人并未达到当前位置，而是在达到该位置前改变运动方向，平滑过渡并前往下一个目

标位置。

运动指令中停止点使用参数 fine，飞越点使用参数“*z*”。“*z*”表示转弯半径，后面使用具体的数值，数值越大转弯半径越大（如 z10、z50 分别表示转弯半径为 10 mm、50 mm）。ABB 机器人系统预定义转弯半径数据如表 3-26 所示。

表 3-26　预定义转弯半径数据

路径区域			
名称	TCP 路径/mm	方向/mm	外轴/mm
z0	0.3	0.3	0.3
z1	1	1	1
z5	5	8	8
z10	10	15	15
z15	15	23	23
z20	20	30	30
z30	30	45	45
z40	40	60	60
z50	50	75	75
z60	60	90	90
z80	80	120	120
z100	100	150	150
z150	150	225	225
z200	200	300	300

（2）z0 与 fine 的区别

由表 3-26 可知 z0 也有转弯半径，但转弯半径很小，为 0.3 mm，并不是 0，而 fine 转弯半径为 0。

转弯区数据为 z 时，系统会预读下一条指令，根据下一条运动指令提交规划过渡路径，实际执行的效果不会精确经过当前点位，而是平滑过渡，机器人没有停顿，因此 TCP 运动的平滑性更好。

转弯区数据为 fine 时，系统不会预读程序，等当前指令运行结束后，程序指针才跳到下一条指令。所以执行 fine 时，机器人会精确到达目标位置，且有短暂停顿，但人眼可能分辨不了。

当运动指令后面使用 I/O 指令时，如果使用 z0，则程序预读后会提前执行 I/O 指令，而使用 fine 则当 TCP 完全到达目标位置后才会执行 I/O 指令。如果 I/O 指令用于控制夹具夹取物料等动作时，转弯区数据使用 z0 时，可能会出现 TCP 到达目标位置前已经完成了夹紧动作，导致夹取失败。

因此需要根据实际情况灵活选择使用 z0 与 fine。

思考与练习

1. 简述 Smart 组件的功能。

2. 已知变量 score 与 grade，利用 if 指令编程实现如下功能：

当 score≥90 时，grade 的值为 A；

当 80≤score≤89 时，grade 的值为 B；

当 70≤score≤79 时，grade 的值为 C；

当 60≤score≤69 时，grade 的值为 D；

当 score<60 时，grade 的值为 E。

3. 利用 test 指令编程实现第 2 题的功能。

4. 利用 for 指令计算 1+2+……+100 的和。

5. 利用 while 指令计算 2+4+……+100 的和。

6. 将本项目的搬运顺序调整为 12→1，同样保持位置一一对应，请编程实现。

7. 将本项目的物料位对应规则调整为 1—12、2—11、……、12—1，即取料盘上从 1 号位开始拾取，放料盘上则从 12 号位开始放置，搬运顺序根据取料盘从 1→12 依次搬运，请编程实现。