English version of the README -> please click here

0.Changelog

- V4.6.4-2025-0822 新增DOGroupDEC、DIGroupDEC、getDOGroupDEC命令。
- V4.6.x-2025-07022 补充关于轨迹拟合指令MovS 指令的说明，以及由此产生的 **GetStartPose** 指令的修改
- V4.6.4-2025-06-04 新增传送带命令 CnvInit/GetCnvObject/StartSyncCnv/CnvMovL/CnvMovC/StopSyncCnv/SetCnvPointOffset/SetCnvTimeCompensation。
- V4.6.4-2025-06-04 新增StartRTOffset、EndRTOffset、OffsetPara命令。
- V4.6.4-2025-05-08 调整圆弧指令接口的运动模式可选参数说明，新增ArcIO指令。
- V4.6.2-2025-05-08 新增力传感器碰撞检测的接口 SetFCCollision 和 FCCollisionSwitch 的指令说明。
- V4.6.0-2024-12-17 调整轨迹复现的接口 **StartPath** 的参数说明，增加sample 和 freq 的描述。
- V4.6.0-2024-12-09 调整关于DO/DI/ToolDO/ToolDI/ToolAI 指令的 index索引范围的描述。
- V4.6.0-2024-11-27 补充指令 CreateTray() 和 GetTrayPoint()的说明。
- V4.6.0-2024/11/25 ServoP()、ServoJ()指令调整点位的运行时间t参数范围为：[0.004,3600.0]
- V4.6.0-2024/11/11 新增license过期后的TCP指令返回固定错误码 -8 的描述。
- V4.6.0-2024/07/20 新增力控运动相关指令：FCForceMode()、FCSetDeviation()、FCSetForceLimit()、FCSetMass()、FCSetStiffness()、FCSetDamping()、FCOff()、FCSetForceSpeedLimit()、FCSetForce()
- V4.6.0-2024/04/24 新增错误码-7："脚本暂停情况下禁止TCP指令"，新增各状态下允许执行的TCP指令列表
- V4.6.0-2024/04/22 支持轨迹恢复功能：点动改为立即指令，新增RunTo、PathRecovery、SetPathRecovery、GetPathRecovery指令
- V4.6.0-2024/04/07 新增力控相关的指令和实时反馈值，RobotMode()指令扩展：力控拖拽模式对外返回拖拽模式，StopDrag()指令扩展，若处于力控拖拽状态则退出力控拖拽
- V4.6.0-2024/03/27 增加RequestControl命令，实时数据增加安全IO及安全状态定义说明
- V4.5.1-2024/02/27 新增RelJoint、RelPointUser、RelPointTool命令
- V4.5.1-2024/02/28 MoveJog指令新增错误返回码-6，MoveJog运动轴与运动类型不匹配

- V4.5.1-2024/02/02 补充 参数分解异常导致的错误码差异的描述，详见文档附录的错误码部分。
- V4.5.1-2024/01/24 SetTool/SetUser的 新增可选参数type ,用来区分是否将坐标系更新全局生效。调整index的范围 [0,50] -> [1,50]
- V4.5.1-2024/01/11 新增用户日志导出指令，新增点位校验检查指令，实时反馈端口新增当前日志导出状态数据
- V4.5.1-2023/12/19 修改点位信息的rx/ry/rz参数范围为：rx/ry/rz(-1.7976931348623157 × 10^(308),1.7976931348623157 × 10^(308))
- V4.5.0-20203/11/24:部分队列指令实例返回值添加队列id
- V4.5.0-2023/11/22：

  1）新增脚本运行过程中禁止执行队列指令规则，其中允许指令有：SpeedFactor()、RobotMode()、DOInstant()、ToolDOInstant()、AOInstant()、Stop()、Pause()、Continue()、GetStartPose()、PositiveKin()、InverseSolution()、GetAngle()、GetPose()、EmergencyStop()、ModbusRTUCreate（）、ModbusCreate()、ModbusClose()、GetInBits()、GetInRegs()、GetCoils()、SetCoils()、GetHoldRegs()、SetHoldRegs()、GetErrorID()、DI()、ToolDI()、AI()、ToolAI()、DIGroup()、GetDO()、GetAO()、GetDOGroup()、SetTool485()、SetToolPower()、SetToolMode()、CalcUser()、CalcTool()、GetInputBool()、GetInputInt()、GetInputFloat()、GetOutputBool()、GetOutputInt()、GetOutputFloat()、SetOutputBool()、SetOutputInt()、

  SetOutputFloat()、GetCurrentCommandId()；其余指令禁止执行

  2）新增对应错误码，-5：当前处于脚本运行/暂停过程中，拒绝执行队列指令，返回-5
- V4.5.0-2023/08/23：增加命令ServoJ、ServoP
- V4.3.0-2023/06/08：增加实时数据CurrentCommandId,VibrationDisZ
- V4.2.9-2023/05/05：删除StopRobot命令
- V4.2.8-2023/04/10：新增GetInputBool、GetInputInt、GetInputFloat、GetOutputBool、GetOutputInt、GetOutputFloat、SetOutputBool、SetOutputInt、SetOutputFloat命令
- V4.2.7-2023/03/29： 1）用户/工具坐标系相关参数类型自定义模式去除 对应指令：User、Tool、MovJIO、MovLIO、RelJointMovJ、RelMovLUser、RelMovLTool、RelMovJUser、RelMovJTool、Circle、Arc、MovL、MovJ、StartPath、PositiveKin、InverseKin 2）坐标系索引范围对应更新到0~50
- V4.2.6-2023/03/21：运动相关指令可选参数vt参数名称更改为speed
- V4.2.5-2023/03/20：更新AO()、AOInstant()指令valuse值参数范围
- V4.2.4-2023/03/16：更新TCP实时反馈数据并注明去除或不实现的字段
- V4.2.3-2023/03/18：TCP新增Pause，Continue，Stop、SetBackDistance、SetPostCollisionMode。删除PauseScript，ContinueScript，StopScript
- V4.2.1-2023/03/14：TCP新增修改用户坐标系、修改工具坐标系、计算用户坐标系、计算 工具坐标系指令:SetUser()、SetTool()、CalcUser()、CalcTool()
- V4.2.0-2023/03/7：TCP新增LUA已有指令：SetToolPower()、SetToolMode()
- V4.1.9-2023/02/28：TCP新增指令文档补充：GetDO()、GetAO ()、GetToolDO ()、GetDOGroup ()、SetTool485()、SetSafeWallEnable ()
- V4.1.8-2023/02/21：运动相关指令新增可选参数vt
- V4.1.7-2023/02/15：调整GetPose()指令参数个数限定，仅允许0/2个参数

- V4.1.6-2023/01/18：新增关于中文字符发送的注意事项：需将发送端的编码方式选择为UTF-8格式，否则会导致中文字符接收异常
- V4.1.5-2023/01/04：EnableRobot()新增负载检查可选参数，SetPayload()新增可根据负载名称进行设置参数类型
- V4.1.4-2022/12/22：TCP实时反馈数据中新增：FA、J4、J5、J6接近暂停状态
- V4.1.3-2022/12/14：RobotMode()新增检测到碰撞状态:ROBOT_MODE_COLLISION；实时反馈数据中新增：检测到碰撞状态RobotCollision；ClearError()指令可清除碰撞状态
- V4.1.2-2022/12/13：安全皮肤的相关指令新增：EnableSafeSkin(status) 、SetSafeSkin(part, status)
- V4.1.1-2022/11/25：轨迹复现指令名称修正：StartPath()
- V4.1.0-2022/11/23: RobotMode()新增初始化状态：ROBOT_MODE_INIT，控制柜启动过程中状态返回ROBOT_MODE_INIT，启动完成且上电成功返回 ROBOT_MODE_DISABLED
- V4.0.9-2022/11/9: 去除MovJ、MovJIO、RelMovJTool、RelMovJUser指令的可选参数r，关节运动不支持r参数的设置。
- V4.0.8-2022/10/24:

  1）DO指令新增固定时间后取反功能。

  2）新增开/关安全皮肤指令 ： SetSafeSkin()

  3）新增进入/退出推拽，设置推拽灵敏度指令：StartDrag()、StopDrag()、DragSensivity()
- V4.0.7-2022/10/08: 补充说明带名称参数格式/范围错误的错误码返回
- V4.0.6-2022/09/22: 补充说明运动指令的坐标系使用规则
- V4.0.5-2022/09/21: 对应错误状态新增可执行指令，处理规则：

1）错误状态可执行指令：错误清除、错误查询、急停、机器人状态；

2）急停状态可执行指令：错误清除、错误查询、急停松开、机器人状态

3）下电状态可执行指令：错误清除、错误查询、上电、机器人状态、急停松开

- V4.0.4-2022/09/19: 新增错误码类型，并对所有指令添加错误状态防护，包括：当前为错误状态，当前为下电状态，当前为急停拍下状态，需先解除这些状态再进行指令的实际处理。
- V4.0.3-2022/09/13:

1）增加查找算法队列指令： GetCurrentCommandID()；

2）算法指令添加对应算法id返回值，包括：DO()、ToolDO()、AO()、MovJ()、MovL()、MovJIO()、Circle()、RelMovJTool()、RelMovLTool()、 RelMovLUser()、RelJointMovJ()；

- V4.0.2-2022/09/2: 新增错误码类型列表；
- V4.0.1-2022/08/23:

1）TCP对应指令名称更新，包括：DOInstant()、ToolDOInstant()、AOInstant()、VelJ()、VelL()、GetStartPose()、StartTrace()、 PositiveKin()、StopRobot()、SetPayload()；

2）TCP部分指令去除，包括：ResetRobot()、HandleTrajPoints()、GetSixForceData()、StartDrag()、StopDrag()、 SetCollideDrag()、SetTerminalKeys()、SetTerminal485()、Arch()、LoadSwitch()、SetArmOrientation()、ServoJ()、ServoP()、StartPath()、StartFCTrace()、Sync()、JointMovJ()、SetSafeSkin()；

3） TCP部分指令参数格式更新,包括：User()、Tool()、PositiveKin()、InverseKin()、MovJ()、MovL()、MovLIO()、MovJIO()、RelMovJTool()、RelMovJUser()、RelMovLUser()、RelJointMovJ()、Arc()、

Circle();

4）RobotMode()，机器人状态对应含义更新；

1. 综述

越疆协作机器人支持TCP/IP二次开发方式：需先在上位机将机器人控制方式设置为"**TCP/IP二次开发模式**"，后连接机器人TCP服务端进行指令发送及对应数据监

控，**TCP/IP控制模式**；具体控制方式详见《Dobot Pro软件使用说明->设置->远程控制章节中；

描述机器人有几种控制方式，此方式针对远程控制机器人；由于基于TCP/IP的通讯具有成本低、可靠性高、实用性强、性能高等特点；许多工业自动化项目对支持TCP/IP协议控制机器人需求广泛，因此CR机器人将设计在TCP/IP协议的基础上，提供了丰富的接口用于与外部设备的交互；

根据设计，CR机器人会开启**29999**、**30004**、**30005**以及**30006**服务器端口；

29999服务器端口(以下简称Dashboard端口)通过一发一收的方式负责接收一些设置以及运动控制相关的指令，即**控制端口接收到客户端约定消息格式后会将结**

果反馈客户端；

30004服务器端口(以下简称实时反馈端口)**每8ms反馈机器人的信息；**

30005服务器端口**每200ms反馈机器人的信息**，

30006端口为**可配置**的反馈机器人信息端口(默认为每**50ms**反馈)；

2.消息格式

TCPIP远程控制命令**不区分大小写格式**；如ENABLEROBOT()/enablerobot()/eNabLErobOt()，控制器都会按照**使能**的命令执行；

消息命令与消息应答都是 ASCII 码格式(字符串形式)。

2.1 命令格式

```
命令名称(Param1,Param2,Param3,……Paramn)
```

命令格式如上所示，由一个命令名称，括号内由参数组成， 每一个参数之间以英文逗号 ”，” 相隔，一个完整的命令以右括号结束。

命令区分为队列指令和立即指令，队列指令和立即指令的区别，详见指令列表中的指令类型

- 队列指令：指令下发后不可立即执行，需发送给算法队列，由算法规划执行执行
- 立即指令：指令下发后立即执行

  注：队列/立即指令详见下表指令类型

2.2 返回格式

2.2.1 返回：

返回命令格式： "指令检查值,{命令返回值},原始命令;"

"ErrorID,{value,...,valuen},命令名称(Param1,Param2,Param3……Paramn);"

命令格式如上所示：

- ErrorID为0时表示命令接收成功；返回非0则代表命令有错误，具体的错误描述见第五章节;
- {value1,value2,value3,...,valuen}表示返回值，没有返回值则返回{}；
  - 队列指令value返回值指队列ID:commandID
  - 立即指令valude返回值为要返回的数据内容
- 命令名称(Param1,Param2,Param3……Paramn)指下发的指令内容。

例：

MovL(pose={-500,100,200,150,0,90})

- 返回：0,{1},MovL(pose={-500,100,200,150,0,90}); //0表示接收成功 {1}：commandID

Mov(-500,100,200,150,0,90) //下发不存在的指令

- 报警： -10000,{},Mov(-500,100,200,150,0,90); //-10000表示命令错误

  GetPose(user = 1, tool = 0)

- 返回：0,{-473.0,-141.0,469.0,-180.0,0.0,90.0},GetPose(user = 1, tool = 0);

3.控制端口命令

上位机可以通过29999端口发送命令给机器人，这些命令被称为控制命令。

控制端口命令列表如下：

指令	描述	支持产品	指令类型
EnableRobot	使能机器人	CR/Nova/ED6	立即指令
DisableRobot	下使能机器人	CR/Nova/ED6	立即指令
ClearError	复位，用于清除错误	CR/Nova/ED6	立即指令
SpeedFactor	设置全局速率比	CR/Nova/ED6	立即指令
User	选择已标定的用户坐标系 （笛卡尔空间显示值 实际生效根据点）	CR/Nova/ED6	队列指令
Tool	选择已标定的工具坐标系	CR/Nova/ED6	队列指令
RobotMode	机器人模式	CR/Nova/ED6	立即指令
SetPayload	设置负载	CR/Nova/ED6	队列指令
DO	设置数字量输出端口状态	CR/Nova/ED6	队列指令
DOInstant	设置数字量输出端口状态	CR/Nova/ED6	立即指令

指令	描述	支持产品	指令类型
ToolDO	设置末端数字量输出端口状态	CR/Nova/ED6	队列指令
ToolDOInstant	设置末端数字量输出端口状态	CR/Nova/ED6	立即指令
AO	设置模拟量输出端口状态	CR/Nova/ED6	队列指令
AOInstant	设置模拟量输出端口状态	CR/Nova/ED6	立即指令
AccJ	设置关节加速度比例。	CR/Nova/ED6	立即指令
AccL	设置笛卡尔加速度比例。	CR/Nova/ED6	立即指令
VelJ	设置关节速度比例。	CR/Nova/ED6	立即指令
VelL	设置笛卡尔速度比例。	CR/Nova/ED6	立即指令
CP	运动时设置平滑过渡	CR/Nova/ED6	立即指令
PowerOn	机器人上电	CR/Nova/ED6	立即指令
RunScript	运行脚本	CR/Nova/ED6	立即指令
Stop	停止	CR/Nova/ED6	立即指令
Pause	暂停	CR/Nova/ED6	立即指令
Continue	继续	CR/Nova/ED6	立即指令
EnableSafeSkin	设置安全皮肤开关状态	CR/Nova/ED6	队列指令
SetSafeSkin	设置安全皮肤灵敏度	CR/Nova/ED6	队列指令
GetStartPose	获取轨迹的第一个点位	CR/Nova/ED6	立即指令
StartPath	轨迹复现	CR/Nova/ED6	队列指令
MovS	轨迹拟合	CR/Nova/ED6	队列指令
PositiveKin	正解	CR/Nova/ED6	立即指令
InverseSolution	逆解	CR/Nova/ED6	立即指令

指令	描述	支持产品	指令类型
SetCollisionLevel	设置碰撞等级	CR/Nova/ED6	队列指令
GetAngle	获取关节坐标系下机械臂的实时位姿	CR/Nova/ED6	立即指令
GetPose	获取笛卡尔坐标系下机械臂的实时位姿	CR/Nova/ED6	立即指令
EmergencyStop	急停	CR/Nova/ED6	立即指令
ModbusRTUCreate	创建ModbusRTU主站，并和从站建立连接	CR/Nova/ED6	立即指令
ModbusCreate	创建Modbus主站，并和从站建立连接	CR/Nova/ED6	立即指令
ModbusClose	和Modbus从站断开连接	CR/Nova/ED6	立即指令
GetInBits	读离散输入功能	CR/Nova/ED6	立即指令
GetInRegs	读输入寄存器	CR/Nova/ED6	立即指令
GetCoils	读线圈功能	CR/Nova/ED6	立即指令
SetCoils	写线圈功能	CR/Nova/ED6	立即指令
GetHoldRegs	读保存寄存器	CR/Nova/ED6	立即指令
SetHoldRegs	写保存寄存器	CR/Nova/ED6	立即指令
GetErrorID	获取错误ID	CR/Nova/ED6	立即指令
DI	获取数字量输入端口状态	CR/Nova/ED6	立即指令
ToolDI	获取末端数字量输入端口状态	CR/Nova/ED6	立即指令
AI	获取模拟量输入端口电压值	CR/Nova/ED6	立即指令
ToolAI	获取末端模拟量输入端口电压值	CR/Nova/ED6	立即指令
DIGroup	获取输入组端口状态	CR/Nova/ED6	立即指令
DOGroup	设置数字输出组端口状态	CR/Nova/ED6	队列指令
BrakeControl	抱闸控制	CR/Nova/ED6	立即指令

指令	描述	支持产品	指令类型
StartDrag	进入拖拽	CR/Nova/ED6	立即指令
StopDrag	退出拖拽	CR/Nova/ED6	立即指令
DragSensivity	拖拽灵敏度设置	CR/Nova/ED6	立即指令
GetDO	获取数字输出端口状态	CR/Nova/ED6	立即指令
GetAO	获取模拟量输出端口状态	CR/Nova/ED6	立即指令
GetDOGroup	获取组数字输出端口状态	CR/Nova/ED6	立即指令
SetTool485	设置末端485通讯参数	CR/Nova/ED6	立即指令
SetSafeWallEnable	设置安全墙开关	CR/Nova/ED6	队列指令
SetToolPower	设置末端电源开关	CR/Nova/ED6	立即指令
SetToolMode	设置末端模式	CR/Nova/ED6	立即指令
SetBackDistance	设置碰撞回退距离	CR/Nova/ED6	队列指令
SetPostCollisionMode	设置选择碰撞后是进入状态	CR/Nova/ED6	队列指令
SetUser()	设置用户坐标系	CR/Nova/ED6	立即指令
SetTool()	设置工具坐标系	CR/Nova/ED6	立即指令
CalcUser()	计算用户坐标系	CR/Nova/ED6	立即指令
CalcTool()	计算工具坐标系	CR/Nova/ED6	立即指令
GetInputBool	获取输入寄存器bool数值	CR/Nova/ED6	立即指令
GetInputInt	获取输入寄存器int数值	CR/Nova/ED6	立即指令
GetInputFloat	获取输入寄存器float数值	CR/Nova/ED6	立即指令
GetOutputBool	获取输出寄存器bool数值	CR/Nova/ED6	立即指令
GetOutputInt	获取输出寄存器int数值	CR/Nova/ED6	立即指令

指令	描述	支持产品	指令类型
GetOutputFloat	获取输出寄存器float数值	CR/Nova/ED6	立即指令
SetOutputBool	设置输出寄存器bool数值	CR/Nova/ED6	立即指令
SetOutputInt	设置输出寄存器int数值	CR/Nova/ED6	立即指令
SetOutputFloat	设置输出寄存器float数值	CR/Nova/ED6	立即指令
MovJ	点到点运动，目标点位为笛卡尔点位	CR/Nova/ED6	队列指令
MovL	直线运动，目标点位为笛卡尔点位	CR/Nova/ED6	队列指令
MovLIO	直线运动过程中并行设置数字输出端口的状态，可设置多组	CR/Nova/ED6	队列指令
MovJIO	点到点运动过程中并行设置数字输出端口的状态，可设置多组	CR/Nova/ED6	队列指令
Arc	圆弧运动。需结合其他运动指令完成圆弧运动	CR/Nova/ED6	队列指令
ArcIO	圆弧运动过程中并行设置数字输出端口的状态，可设置多组	CR/Nova/ED6	队列指令
Circle	整圆运动	CR/Nova/ED6	队列指令
MoveJog	关节点动	CR/Nova/ED6	立即指令
RelMovJTool	沿工具坐标系进行相对运动，末端运动 方式为关节运动	CR/Nova/ED6	队列指令
RelMovLTool	沿工具坐标系进行相对运动指令，末端运动方式为直线运动	CR/Nova/ED6	队列指令
RelMovJUser	沿用户坐标系进行相对运动指令，末端运动方式为关节运动	CR/Nova/ED6	队列指令
RelMovLUser	沿用户坐标系进行相对运动指令，末端运动方式为直线运动	CR/Nova/ED6	队列指令
RelJointMovJ	沿各轴关节坐标系进行相对运动指令，末端运动方式为关节运动	CR/Nova/ED6	队列指令
GetCurrentCommandId	获取当前运行指令的队列ID	CR/Nova/ED6	立即指令
CheckOddMovL()	检查两个点位直线运动过程中点位可达性	CR/Nova/ED6	立即指令
CheckOddMovJ()	检查两个点位关节运动过程中点位可达性	CR/Nova/ED6	立即指令
CheckOddMovC()	检查经过指定三个点位的圆弧运动过程中点位可达性	CR/Nova/ED6	立即指令

指令	描述	支持产品	指令类型
LogExportUSB()	控制器日志文件导出	CR/Nova/ED6	立即指令
GetExportStatus()	获取当前日志导出状态	CR/Nova/ED6	立即指令
RelPointUser()	沿用户坐标系笛卡尔点偏移	CR/Nova/ED6	立即指令
RelPointTool()	沿工具坐标系笛卡尔点偏移	CR/Nova/ED6	立即指令
RelJoint()	关节点位偏移	CR/Nova/ED6	立即指令
WeldArcSpeedStart()	焊接过程运动速度以绝对速度规划的启动指令(队列指令)	CR	队列指令
WeldArcSpeedEnd()	焊接过程运动速度以绝对速度规划的停止指令(队列指令)	CR	队列指令
WeldArcSpeed()	不受全局速度影响的单位为毫米/秒的只影响焊接运动的速度指令(队列指令)	CR	队列指令
WeaveStart()	焊接过程运动轨迹按照设置参数的启动指令(队列指令)	CR	队列指令
WeaveEnd()	焊接过程运动轨迹按照设置参数的停止指令(队列指令)	CR	队列指令
WeaveParams()	设置摆焊的参数	CR	队列指令
RelPointWeldLine()	多层多道功能使用的焊接轨迹直线偏移指令	CR	队列指令
RelPointWeldArc()	多层多道功能使用的焊接轨迹圆弧偏移指令	CR	队列指令
EnableFTSensor()	开启/关闭力传感器开关	CR	立即指令
SixForceHome()	将力传感器当前数值置零	CR	立即指令
GetForce()	读取施加在TCP方向上的力和力矩	CR	立即指令
ForceDriveMode()	设置并进入力控拖拽模式	CR	立即指令
ForceDriveSpeed()	力控拖拽百分比设置	CR	立即指令
RunTo	RunTo运动	CR/Nova/ED6	立即指令
PathRecovery	轨迹恢复运动	CR/Nova/ED6	立即指令
GetDOGroupDEC	获取一组数字输出组端口状态，将组DO电平按0/1组成一个二进制数，再转化为十进制输出	CR/Nova/ED6	立即指令

指令	描述	支持产品	指令类型
DOGroupDEC	设置输出组端口状态，给定一个十进制值，将该值转换为二进制值，按bit对应给定输入索引的DO值。	CR/Nova/ED6	队列指令
DIGroupDEC	获取输入组端口状态，将组DI电平按0/1组成一个二进制数，再转化为十进制输出	CR/Nova/ED6	队列指令

指令	描述	支持产品	指令类型
RequestControl	请求将设备控制模式切换为TCP模式	CR/Nova/ED6	立即指令
FCForceMode	以用户指定的配置参数开启力控	CR	队列指令
FCSetDeviation	设置力控模式下的位移和姿态偏差，若力控过程中恒力偏移了较大的距离，机器人进行相应处理	CR	立即指令
FCSetForceLimit	设置最大力限制	CR	立即指令
FCSetMass	设置力惯性	CR	立即指令
FCSetStiffness	设置力刚度	CR	立即指令
FCSetDamping	设置力阻尼	CR	立即指令
FCOff	关闭力控	CR	队列指令
FCSetForceSpeedLimit	力控调节速度限制	CR	立即指令
FCSetForce	实时调整恒力设置	CR	立即指令
CreateTray	创建托盘	CR/Nova/ED6	立即指令
GetTrayPoint	获得托盘内的点位信息	CR/Nova/ED6	立即指令
SetFCCollision	设置力控碰撞检测的阈值参数	CR	立即指令
FCCollisionSwitch	力传感器碰撞检测功能开关	CR	立即指令
StartRTOffset	启动坐标系偏移	CR	队列指令
EndRTOffset	结束坐标系偏移	CR	队列指令
OffsetPara	设置坐标系偏移参数	CR	立即指令

3.1 EnableRobot

- 功能：使能机器人
- 格式：

  EnableRobot()
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值
load	double	负载重量kg	否	上一次全局指令设置值
centerX	double	X方向偏心距离mm	否	上一次全局指令设置值
centerY	double	Y方向偏心距离mm	否	上一次全局指令设置值
centerZ	double	Z方向偏心距离mm	否	上一次全局指令设置值
isCheck	int	是否进行负载检查：1：检查，0：不检查	否	0

- 返回：

  ErrorID,{},EnableRobot();
- 示例：

  EnableRobot() //使能

  EnableRobot(2) //使能，设置负载值为2，偏心值为之前设置的值

  EnableRobot(2,1,2,3)//使能，设置负载值为2，偏心值为1，2，3

  EnableRobot(2,1,2,3,1)//使能，设置负载值为2，偏心值为1，2，3，进行负载检查
- 说明：**可选参数数量：0/1/4/5**

机型	load范围(KG)	centerX范围(mm)	centerY范围(mm)	centerZ范围(mm)	isCheck范围
CR3/CR3A	[0,3]	[-999,999]	[-999,999]	[-999,999]	0/1
CR5/CR5A	[0,5]	[-999,999]	[-999,999]	[-999,999]	0/1
CR7/CR7A	[0,7]	[-999,999]	[-999,999]	[-999,999]	0/1
CR10/CR10A	[0,10]	[-999,999]	[-999,999]	[-999,999]	0/1
CR12/CR12A	[0,12]	[-999,999]	[-999,999]	[-999,999]	0/1
CR16/CR16A	[0,16]	[-999,999]	[-999,999]	[-999,999]	0/1
CR20A	[0,20]	[-999,999]	[-999,999]	[-999,999]	0/1
Nova2	[0,2]	[-999,999]	[-999,999]	[-999,999]	0/1
Nova5	[0,5]	[-999,999]	[-999,999]	[-999,999]	0/1
Magician E6	[0,0.75]	[-999,999]	[-999,999]	[-999,999]	0/1

3.2 DisableRobot

- 功能：下使能机器人
- 格式：

DisableRobot()

- 支持端口：29999
- 返回：

  ErrorID,{},DisableRobot();

- 示例：

  DisableRobot()//下使能

3.3 ClearError

- 功能：清错机器人
- 格式：

  ClearError()

- 支持端口：29999
- 返回：

  ErrorID,{},ClearError();

- 示例：

  ClearError()//机器人清错

- 说明：清除报警后，用户可以根据RobotMode来判断机器人是否还处于报警状态；对于清除不掉的报警需要重启控制柜解决；(详见GetErrorID说明)

3.5 SpeedFactor

- 功能：设置全局速度比例。
- 格式：

  SpeedFactor(ratio)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	取值范围
ratio	int	运动速度比例	是	[1,100]

- 返回：

  ErrorID,{},SpeedFactor(ratio);

- 示例：

  SpeedFactor(80)//设置全局速度比例为80%

3.6 User（队列指令）

- 功能：选择已标定的用户坐标系
- 格式：

  User(index)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	取值范围
index	int	选择已标定的用户坐标系	是	[0,50]

- 返回：ResultID:队列id

  ErrorID,{ResultID},User(index);

- 示例：

  User(1)//设置全局用户坐标系为1

3.7 Tool（队列指令）

- 功能：选择已标定的工具坐标系
- 格式：

  Tool(index)

- 支持端口：29999
- 参数详解：ResultID:队列id

参数名	类型	含义	是否必填	取值范围
index	int	选择已标定的工具坐标系	是	[0,50]

- 返回：

  ErrorID,{ResultID},Tool(index);

- 示例：

  Tool(1)//设置全局工具坐标系为1

3.8 RobotMode

- 功能：机器人状态。
- 格式

  RobotMode()

- 支持端口：29999
- 返回值：

模式	描述	备注
1	ROBOT_MODE_INIT	初始化状态

模式	描述	备注
2	ROBOT_MODE_BRAKE_OPEN	抱闸松开
3	ROBOT_MODE_POWEROFF	本体下电状态
4	ROBOT_MODE_DISABLED	未使能(抱闸未松开)
5	ROBOT_MODE_ENABLE	使能(空闲)
6	ROBOT_MODE_BACKDRIVE	拖拽(关节拖拽/力控拖拽)
7	ROBOT_MODE_RUNNING	运行状态（含脚本和TCP队列运行）
8	ROBOT_MODE_SINGLE_MOVE	单次运动状态(点动)
9	ROBOT_MODE_ERROR	错误状态
10	ROBOT_MODE_PAUSE	暂停状态
11	ROBOT_MODE_COLLISION	碰撞状态

- 返回：

  ErrorID,{Value},RobotMode(); //Value为返回模式值

- 示例：

  RobotMode()//获取当前机器人状态

  注：

  开启抱闸，状态为2；

  本体下使能，状态为4；

  使能成功后，则状态为5 ；

  机器人进入拖拽模式(使能状态)，状态为6；

  机器人运行，状态为7；运行状态包括：轨迹复现/拟合中、机器人运动(movj())以及脚本运行；

  机器人在点动，状态为8；

  机器人暂停，状态为10；

  其中报警优先级最高，其他状态同时存在时，若有报警，先将状态置

3.9 SetPayload（队列指令）

- 功能：设置当前的负载
- 支持端口：29999
- 格式（1）：

  SetPayload(load,x,y,z)

- 参数详解：

参数名	类型	含义	是否必填	默认值
load	double	负载重量 kg	是	
x	double	偏心坐标X mm	否	上一次全局指令设置值

参数名	类型	含义	是否必填	默认值
y	double	偏心坐标Y mm	否	上一次全局指令设置值
z	double	偏心坐标Z mm	否	上一次全局指令设置值

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},SetPayload(load,x,y,z);

- 格式（2）：

  SetPayload(name)

- 参数详解：

参数名	类型	含义	是否必填
name	string	负载名称	是

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},SetPayload(name);

- 示例：

  SetPayload(3) //设置负载值为3，偏心距离为默认值

  SetPayload(3,1,2,3) //设置负载值为3，偏心距离为1，2，3

  SetPayload("Load1"); //设置负载为"Load1"

3.10 DO（队列指令）

- 功能：设置数字输出端口状态
- 格式：

  DO(index,status,time)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	取值范围
index	int	数字输出索引，取值范围：1~MAX或100~1000	是		[1,MAX] [100,1000]
status	int	数字输出端口状态，1：打开；0：关闭	是		0/1
time	int	持续输出时间，单位ms	否	0	[25,60000]

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},DO(index,status);

- 示例：

  DO(1,1,2000)//设置DO1为打开状态，2000ms后关闭

- 说明：使用取值范围100-1000需要有拓展IO模块的硬件支持；

- MAX是代表当前控制柜的 DO范围。不同控制器的DO资源数量不一样。 所以DO的 index的索引范围需要根据控制器类型来分别讨论。 小型控制柜为8个，一代柜 16个 二代柜 24个。

3.11 DOInstant

- 功能：设置数字输出端口状态
- 格式：

  DOInstant(index,status)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	取值范围
index	int	数字输出索引	是	[1,MAX] [100,1000]
status	int	数字输出端口状态，1：高电平；0：低电平	是	0/1

- 返回：

  ErrorID,{},DOInstant(index,status);
- 示例：

  DOInstant(1,1)//设置DO1为打开状态
- 说明：使用取值范围100-1000需要有拓展IO模块的硬件支持；
- MAX是代表当前控制柜的 DO范围。不同控制器的DO资源数量不一样。 所以DO的 index的索引范围需要根据控制器类型来分别讨论。 小型控制柜为8个，一代柜 16个 二代柜 24个。

3.12 ToolDO（队列指令）

- 功能：设置末端数字输出端口状态
- 格式：

  ToolDO(index,status)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	取值范围
index	int	数字输出索引	是	[1,MAX]
status	int	数字输出端口状态，1：打开；0：关闭	是	0/1

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},ToolDO(index,status);
- 示例：

  ToolDO(1,1) //设置末端DO1为打开状态

MAX是代表当前末端的的 DO范围。不同末端的DO资源数量不一样。 CR20A的末端DO为 4个，其余机型为2个。

3.13 ToolDOInstant

- 功能：设置末端数字输出端口状态
- 格式：

  ToolDOInstant(index,status)

- 支持端口：29999
  - 参数详解：

参数名	类型	含义	是否必填	取值范围
index	int	数字输出索引	是	[1,MAX]
status	int	数字输出端口状态，1：打开；0：关闭	是	0/1

- 返回：

  ErrorID,{},ToolDOInstant(index,status);

- 示例：

  ToolDOInstant(1,1)//设置末端DO1为打开状态

  MAX是代表当前末端的的 DO范围。不同末端的DO资源数量不一样。 CR20A的末端DO为 4个，其余机型为2个。

3.14 AO（队列指令）

- 功能：设置控制柜模拟输出端口的模拟值
- 格式：

  AO(index,value)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	取值范围
index	int	模拟输出索引	是	1/2
value	double	对应index的模拟值	是	电压取值范围0~10V，电流取值范围为4~20

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},AO(index,value);

- 示例：

  AO(1,2)//设置模拟输出端口1的模拟值为2

3.15 AOInstant

- 功能：设置控制柜模拟输出端口的模拟值
- 格式：

  AOInstant(index,value)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	取值范围
index	int	模拟输出索引	是	1/2
value	double	对应index的模拟值	是	电压取值范围0~10V，电流取值范围为4~20

- 返回：

  ErrorID,{},AOInstant(index,value);
- 示例：

  AOInstant(1,2)//设置模拟输出端口1的模拟值为2

3.16 AccJ

- 功能：设置关节加速度比例。
- 格式：

  AccJ(R)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	取值范围
R	int	关节加速度百分比	是	[1,100]

- 返回：

  ErrorID,{},AccJ(R);
- 示例：

  AccJ(50) //设置关节加速度比例为50%

3.17 AccL

- 功能：设置笛卡尔加速度比例。
- 格式：

  AccL(R)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	取值范围
R	int	笛卡尔加速度比例	是	[1,100]

- 返回：

  ErrorID,{},AccL(R);

- 示例：

  AccL(50)//设置笛卡尔加速度比例为50%

3.18 VelJ

- 功能：设置关节速度比例。
- 格式：

  VelJ(R)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	取值范围
R	int	关节速度比例	是	[1,100]

- 返回：

  ErrorID,{},VelJ(R);

- 示例：

  VelJ(50)//设置关节速度比例为50%

3.19 VelL

- 功能：设置笛卡尔速度比例。
- 格式：

  VelL(R)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	取值范围
R	int	笛卡尔速度比例	是	[1,100]

- 返回：

  ErrorID,{},VelL(R);

- 示例：

  VelL(50)//设置笛卡尔速度比例为50%

3.20 CP

- 功能：设置CP比例。CP即平滑过渡，机械臂从起始点经过中间点到达终点时，经过中间点是以直角方式过渡还是以曲线方式过渡。
- 格式：

  CP(R)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	取值范围
R	int	平滑过渡比例	是	[0,100]

- 返回：

  ErrorID,{},CP(R);
- 示例：

  CP(50)//设置平滑过渡比例为50%

3.21 PowerOn

- 功能：机器人上电。
- 格式：

  PowerOn()
- 支持端口：29999
- 返回：

  ErrorID,{},PowerOn();
- 示例：

  PowerOn()//上电
- 说明：机器人上电到完成，需要等待大概10秒钟的时间接口才返回

3.22 RunScript

- 功能：运行脚本。
- 格式：

  RunScript(projectName)
- 支持端口：29999
- 参数详解：

参数名	类型	含义
projectName	string	脚本名称

- 返回：

  ErrorID,{},RunScript(projectName);

- 示例：

  RunScript("123")

  注：需将发送端的编码方式选择为UTF-8格式，否则会导致中文字符接收异常

  若为纯数字工程名称，则需要添加""，参数格式为字符串类型

  如果需要运行脚本后立即暂停，则需要在RunScript指令后至少间隔1s再下发 pause

3.23 Stop

- 功能：停止运动(或脚本)。
- 格式：

  Stop()

- 支持端口：29999
- 返回：

  ErrorID,{},Stop();

- 示例：

  Stop()

  注：可以停止点动、脚本运行、关节运动等一系列运动

3.24 Pause

- 功能：暂停运动(或脚本)。
- 格式：

  Pause()

- 支持端口：29999
- 返回：

  ErrorID,{},Pause();

- 示例：

  Pause()

  注：可以暂停movj()等一系列运动，使机器人处于暂停状态，点动不可暂停

3.25 Continue

- 功能：继续运动(或脚本)。
- 格式：

  Continue()
- 支持端口：29999
- 返回：

  ErrorID,{},Continue();
- 示例：

  Continue()

  注：继续运动，暂停状态下的算法队列指令可继续运动，机器人处于运行状态

3.26 PositiveKin

- 功能：正解。（给定机器人各关节的角度，计算出机器人末端的空间位置）
- 格式：

  PositiveKin(J1,J2,J3,J4,J5,J6,user=1,tool=1)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	取值范围
J1	double	J1 轴位置，单位：度	是		任意值
J2	double	J2 轴位置，单位：度	是		任意值
J3	double	J3 轴位置，单位：度	是		任意值
J4	double	J4 轴位置，单位：度	是		任意值
J5	double	J5 轴位置，单位：度	是		任意值
J6	double	J6 轴位置，单位：度	是		任意值
user=1	string	选择已标定的用户坐标系	否	上一次全局指令设置值	[0,50]
tool=1	string	选择已标定的工具坐标系	否	上一次全局指令设置值	[0,50]

- 返回：

  **ErrorID,{x,y,z,a,b,c},PositiveKin(J1,J2,J3,J4,J5,J6,user=1,tool=1); **

  {x,y,z,a,b,c}指返回的空间位置
- 示例：

  PositiveKin(0,0,-90,0,90,0,user=1,tool=1) //下发关节角度返回当前的机器人末端的空间位置

  返回：

  0,
  {473.000000,-141.000000,469.000000,-180.000000,-0.000000,-90.000000},PositiveKin(0,0,-90,0,90,0,,user=1,tool=1);

3.27 InverseKin

- 功能：逆解。（已知机器人末端的位置和姿态，计算机器人各关节的角度值）
- 格式：

  InverseKin(X,Y,Z,Rx,Ry,Rz,User,Tool,useJointNear,JointNear)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	取值范围
X	double	J1 轴位置，单位：度	是		任意值
Y	double	J2 轴位置，单位：度	是		任意值
Z	double	J3 轴位置，单位：度	是		任意值
Rx	double	J4 轴位置，单位：度	是		任意值
Ry	double	J5 轴位置，单位：度	是		任意值
Rz	double	J6 轴位置，单位：度	是		任意值
useJointNear =1	string	是否角度选解 1:jointNear数据有效; 0:jointNear数据无效	否	0	0/1
jointNear={j1,j2,j3,j4,j5,j6}	string	选解六个关节角度值	否	{0,0,0,0,0,0}	任意值
user=1	string	选择已标定的用户坐标系	否	上一次全局指令设置值	[0,50]
tool=1	string	选择已标定的工具坐标系	否	上一次全局指令设置值	[0,50]

- 返回值：

  ErrorID,{J1,J2,J3,J4,J5,J6},InverseKin(X,Y,Z,RX,RY,RZ,useJointNear =1,jointNear={J1,J2,J3,J4,J5,J6},user=1,tool=1);

  {J1,J2,J3,J4,J5,J6}:逆解的关节值
- 示例：

  InverseKin(473.000000,-141.000000,469.000000,-180.000000,0.000,-90.000,useJointNear =1,jointNear={0,0,-90,0,90,0})

  返回：

  0,{0,0,-90,0,90,0},InverseKin(473.000000,-141.000000,469.000000,-180.000000,0.000,-90.000,useJointNear =1,jointNear={0,0,-90,0,90,0})

  注：若jointNear参数未带，仅带useJointNear参数，则useJointNear参数无效

3.28 SetCollisionLevel（队列指令）

- 功能：设置碰撞等级。
- 格式：

  SetCollisionLevel(level)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	取值范围
level	int	level: 碰撞等级 0：关闭碰撞检测 1~5：等级越高越灵敏	是	[0,5]

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},SetCollisionLevel(level);
- 示例：

  SetCollisionLevel(1)//设置碰撞等级为1

3.29 GetAngle

- 功能：获取关节坐标系下机械臂的实时位姿
- 格式：

  GetAngle()
- 支持端口：29999

  返回：

  **ErrorID,{J1,J2,J3,J4,J5,J6},GetAngle(); **

  {J1,J2,J3,J4,J5,J6}表示当前位置的关节坐标值;
- 示例：

  GetAngle()//获取当前位置的关节坐标值

  返回：

  0,{0.0,0.0,90.0,0.0,-90.0,0.0},GetAngle();

3.30 GetPose

- 功能：获取笛卡尔坐标系下机械臂的实时位姿(如果设置了用户坐标系或工具坐标系，则获取的位姿为设置坐标系下的位姿)
- 格式：

  GetPose()
- 支持端口：29999
- 参数详解：0/2

参数名	类型	含义	是否必填	默认值	取值范围
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50]
tool=1	string	工具坐标系索引，取值范围：0-50，若不带则为之前设置的全局工具坐标系	否	上一次全局指令设置值	[0,50]

- 返回：

  ErrorID,{X,Y,Z,Rx,Ry,Rz},GetPose();

  {X,Y,Z,Rx,Ry,Rz}表示当前位置的笛卡尔坐标值；

- 示例：

  GetPose() //获取当前用户、工具坐标系下的位姿值

  返回：

  0,{-473.0,141.0,469.0,-180.0,0.0,90.0},GetPose();

- 示例：

  GetPose(user = 1, tool = 0)//获取用户坐标系1，工具坐标系0下的位姿值

  返回：

  0,{-473.0,-141.0,469.0,-180.0,0.0,90.0},GetPose(user = 1, tool = 0);

  注：可选参数可全不带，也可全带，不可只带其中一个

3.31 EmergencyStop

- 功能：急停
- 格式：

  EmergencyStop(1)

- 支持端口：29999
- 参数详解：
参数名	类型	含义	是否必填	取值范围
value	int	1:按下急停 0：松开急停	是	0/1

- 返回：

  ErrorID,{},EmergencyStop(1);

- 示例：

  EmergencyStop(1)//按下急停

3.32 ModbusRTUCreate

- 功能：创建modbusRTU主站。

- 格式：

  ModbusRTUCreate(slave_id,baud,parity,data_bit,stop_bit)

- 支持端口：29999

- 参数详解：

参数名	类型	含义	是否必填	默认值	取值范围
slave_id	int	从站id；	是		
baud	int	波特率	是		
parity	string	校验位, "N": 无校验, "O": 奇校验, "E": 偶校验	否	E	N/O/E
data_bit	int	数据位	否	8	
stop_bit	int	停止位	否	1	

- 返回：

  ErrorID,{index},ModbusRTUCreate((slave_id,baud,parity,data_bit,stop_bit);

  {index}: 返回的主站索引，最多支持5个设备,取值范围(0~4)；

- 示例：

  ModbusRTUCreate(1, 115200, "E", 8, 1)

3.32 ModbusCreate

- 功能：创建modbus主站。

- 格式：

  ModbusCreate(ip,port,slave_id,isRTU)

- 支持端口：29999

- 参数详解：

参数名	类型	含义	是否必填	默认值	取值范围
ip	string	从站ip地址；	是		
port	int	从站端口;	是		
slave_id	int	从站ID)	是		大于等于0的整数
isRTU	int	建立modbus TCP/RTU通信 0：建立modbusTCP通信； 1：建立modbusRTU通信；	否	0	0/1

- 返回：

  ErrorID,{index},ModbusCreate(ip,port,slave_id,isRTU);

  {index}: 返回的主站索引，最多支持5个设备,取值范围0~4;

- 示例：建立RTU通信主站(60000末端透传端口)

  ModbusCreate("127.0.0.1",60000,1,0)

3.33 ModbusClose

- 功能：和Modbus从站断开连接,释放主站。
- 格式：

  ModbusClose(index)
- 支持端口：29999
- 参数详解：

参数名	类型	含义
index	int	返回的主站索引；

- 返回：

  ErrorID,{},ModbusClose(index);
- 示例：

  ModbusClose(0)

3.34 GetInBits

- 功能：读离散输入功能。
- 格式：

  GetInBits(index,addr,count)
- 支持端口：29999
- 参数详解：

参数名	类型	含义
index	int	返回的主站索引；
addr	int	视从站配置而定；
count	int	个数，取值范围1~16；

- 返回：

  ErrorID,{value1,value2,...,valuen},GetInBits(index,addr,count);

  {value1,value2...,valuen}：按位获取结果
- 示例：

  GetInBits(0,3000,5)

  返回：

  0,{1,0,1,1,0},GetInBits(0,3000,5);

3.35 GetInRegs

- 功能：读输入寄存器。
- 格式：

  GetInRegs(index,addr,count,valType)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
index	int	返回的主站索引；	是		
addr	int	视从站配置而定；	是		
count	int	个数，取值范围：1 -4	是		[1,4]
valType	string	数据格式 U16: 读取16位无符号数（2个字节，占用1个寄存器） U32: 读取32位无符号数（4个字节，占用2个寄存器） F32: 读取32位浮点数（4个字节，占用2个寄存器） F64: 读取64位浮点数（8个字节，占用4个寄存器）	否	U16	U16/U32/F32/F64

- 返回：

  ErrorID,{value1,value2,...,valuen},GetInRegs(index,addr,count,valType);

  {value1,value2...,valuen}：按变量类型返回
- 示例：

  GetInRegs(0,4000,3)

  正常返回：

  0,{5,18,12},GetInRegs(0,4000,3);

  错误返回：

  -1,{},GetInRegs(0,4000,3);

3.36 GetCoils

- 功能：读线圈功能。
- 格式：

  GetCoils(index,addr,count)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	返回的主站索引；	是	

参数名	类型	含义	是否必填	参数范围
addr	int	视从站配置而定；	是	
count	int	个数	是	[1,16]

- 返回：

  ErrorID,{value1,value2,...,valuen},GetCoils(index,addr,count);

  {value1,value2...,valuen}：按变量类型返回

- 示例：

  GetCoils(0,1000,3)

  正常返回：

  0,{1,1,0},GetCoils(0,1000,3);

  错误返回：

  -1,{},GetCoils(0,1000,3);

3.37 SetCoils

- 功能：写线圈功能。
- 格式：

  SetCoils(index,addr,count,valTab)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	返回的主站索引；	是	
addr	int	视从站配置而定；	是	
count	int	个数	是	[1,16]
valTab	string	写线圈地址值	是	

- 返回：

  ErrorID,{},SetCoils(index,addr,count,valTab);

- 示例：

  SetCoils(0,1000,3,{1,0,1})

  正常返回：

  0,{},SetCoils(0,1000,3,{1,0,1});

  错误返回：

  -1,{},SetCoils(0,1000,3,{1,0,1});

3.38 GetHoldRegs

- 功能：读保持寄存器。
- 格式：

  GetHoldRegs(index,*addr*, *count*,valType)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认参数	参数范围
index	int	index,返回的主站索引，最多支持5个设备	是		[0,4]
addr	int	保持寄存器的起始地址	是		视从站配置而定
count	int	读取指定数量type类型的数据	是		
valType	string	数据类型： U16：读取16位无符号整数（2个字节，占用1个寄存器） U32：读取32位无符号整数（4个字节，占用2个寄存器） F32：读取32位单精度浮点数（4个字节，占用2个寄存器） F64：读取64位双精度浮点数（8个字节，占用4个寄存器）	否	U16	U16/U32/F32/F64

- 返回：

  ErrorID,{value1,value2,...,valuen},GetHoldRegs(index,*addr*, *count*,valType);

  {value1,value2...,valuen}：按变量类型返回
- 示例：从地址3095开始读取一个16位无符号整数

  GetHoldRegs(0,3095,1)

  正常返回：

  0,{13},GetHoldRegs(0,3095,1);

  错误返回：

  -1,{},GetHoldRegs(0,3095,1);

3.39 SetHoldRegs

- 功能：写保存寄存器。
- 格式：

  SetHoldRegs(index,*addr*, *count*,*valTab*,valType)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认参数	参数范围
index	int	index,返回的主站索引，最多支持5个设备,取值范围(0~4)	是		[0,4]
addr	int	保持寄存器的起始地址。视从站配置而定；	是		
count	int	写入指定数量type类型的数据。取值范围：1~4	是		
valTab	string	保持寄存器地址的值	是		
valType	string	数据类型 U16：读取16位无符号整数（2个字节，占用1个寄存器） U32：读取32位无符号整数（4个字节，占用2个寄存器） F32：读取32位单精度浮点数（4个字节，占用2个寄存器） F64：读取64位双精度浮点数（8个字节，占用4个寄存器）	否	U16	U16/U32/F32/F64

- 返回：

  ErrorID,{},SetHoldRegs(index,*addr*, *count*,*valTab*,valType);

- 示例：从地址3095开始，写入两个16位无符号整数值6000，300

  SetHoldRegs(0,3095,2,{6000,300}, U16)

  正常返回：

  0,{},SetHoldRegs(0,3095,2,{6000,300}, U16);

  错误返回：

  -1,{},SetHoldRegs(0,3095,2,{6000,300}, U16);

3.40 GetErrorID

- 功能：获取机器人错误码
- 格式：

  GetErrorID()

- 支持端口：29999
- 返回：

  ErrorID,{[[id,...,id], [id], [id], [id], [id], [id], [id]]},GetErrorID();

  [id, ..., id]为控制器以及算法报警信息，后面六个[id]分别表示六个伺服的报警信息；

- 示例：

  GetErrorID()

  返回：

0,{[[],[],[],[],[],[],[]],GetErrorId();

- 说明：对于错误码对应的错误内容请参考控制器错误描述文件alarm_controller.json以及伺服错误描述alarm_servo.json；

3.41 DI

- 功能：获取数字量输入端口状态
- 格式：

  DI(index)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	数字输入索引	是	[1,MAX] [100,1000]

- 返回：

  ErrorID,{value},DI(index);

  {value}：index的值状态，取值范围0/1;

- 示例：

  DI(1)//返回数字输入端口1为低电平：

  返回：

  0,{0},DI(1);

- 说明：使用取值范围100-1000需要有拓展IO模块的硬件支持；

  参数范围中的MAX 根据控制柜类型不同而不同。 一代控制柜MAX=32,其中DI17开始是 DO的输入复用。

  二代控制柜 MAX=32, 小型控制柜MAX为8, ,MG6 也是8 。

3.42 ToolDI

- 功能：获取末端数字量输入端口状态
- 格式：

  ToolDI(index)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	末端数字量输入索引	是	[1,MAX]

  index参数中的MAX代表最大的索引值，CR20A 的MAX为4 ， 其余机型为2.

- 返回：

ErrorID,{value},ToolDI(index);

- 示例：

  ToolDI(2)

  返回末端数字输入端口2为高电平：

  0,{1},ToolDI(2);

3.43 AI

- 功能：获取模拟量输入端口模拟值（立即指令）
- 格式：

  AI(index)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	控制柜模拟输入索引	是	1/2

- 返回：

  ErrorID,{value},AI(index);

  {value}：index的值状态，取值范围0/1;

- 示例：

  AI(2)

  返回：模拟输入端口2的模拟值为3.5：

  0,{3.5},AI(2);

3.44 ToolAI

- 功能：获取末端模拟量输入端口模拟值
- 格式：

  ToolAI(index)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	末端模拟输入索引	是	[1,MAX]

MAX : 是最大的ToolAI 索引，目前CR20A MAX=4 ,其余机型MAX=2

- 返回：

  ErrorID,{value},ToolAI(index);

{value}：index的值状态，取值范围0/1;

- 示例：

  ToolAI(1)

  返回：模拟输入端口1的电压值为1.5V：

  0,{1.5},ToolAI(1);

3.45 DIGroup

- 功能：获取输入组端口状态
- 格式：

  DIGroup(index1,index2,...,indexn)

- 参数数量：不固定(最大支持64个)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	参数范围
index1	int	数字输入索引	[1,MAX] [100,1000]
...	...	...	...
indexn	int	数字输入索引	[1,MAX] [100,1000]

- 返回：

  ErrorID,{value1,value2,...,valuen},DIGroup(index1,index2,...,indexn);

  {value1,...valuen,}：返回当前index1到indexn的电压值;

- 示例：

  DIGroup(4,6,2,7)

  返回：获取的输入4、6、2、7端口的电平分别为1，0，1，1

  0,{1,0,1,1},DIGroup(4,6,2,7);

- 说明：使用取值范围100-1000需要有拓展IO模块的硬件支持；

  参数范围中的MAX 根据控制柜类型不同而不同。 一代控制柜MAX=32,其中DI17开始是 DO的输入复用。

  二代控制柜 MAX=32, 小型控制柜MAX为8 ,MG6 也是8 。

3.46 DOGroup（队列指令）

- 功能：设置输出组端口状态
- 格式：

  DOGroup(index1,value1,index2,value2,...,indexn,valuen)

- 参数数量：不固定(最大支持64个)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	参数范围
index1	int	设置数字输出索引	[1,MAX] [100,1000]
value1	int	设置数字输出端口状态	0/1
...	...	...	...
indexn	int	设置数字输出索引	[1,MAX] [100,1000]
valuen	int	设置数字输出端口状态	0/1

- 返回： ResultID:算法队列ID

  ErrorID,{ResultID},DOGroup(index1,value1,index2,value2,...,indexn,valuen);

- 示例：

  DOGroup(4,1,6,0,2,1,7,0)

  返回：分别设置输出端口4、6、2、7分别为1、0、1、0

  0,{ResultID},DOGroup(4,1,6,0,2,1,7,0);

- 说明：使用取值范围100-1000需要有拓展IO模块的硬件支持；

  参数范围中的MAX 根据控制柜类型不同而不同。 一代控制柜MAX=16,二代控制柜 MAX=24, 小型控制柜MAX为8 ,MG6 也是8 。

3.47 BrakeControl

- 功能：开关抱闸
- 格式：

  BrakeControl(axisID,value)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
axisID	int	关节轴号	是	[1,6]
value	int	设置抱闸状态； 0:关闭抱闸 1：打开抱闸	是	0/1

- 返回：

  ErrorID,{},BrakeControl(axisID,value);

- 示例：打开关节1抱闸

  BrakeControl(1,1)

- 返回：

  0,{},BrakeControl(1,1);

  注：抱闸的控制需要机器人在下使能的条件下进行；否则机器人错误返回-1；

3.48 StartDrag

- 功能：进入拖拽(在报错状态下，不可进入拖拽)
- 格式：

  StartDrag()
- 支持端口：29999
- 返回：

  ErrorID,{},StartDrag();
- 示例：

  StartDrag()//进入拖拽

3.49 StopDrag

- 功能：退出拖拽（在报错状态下，可以退出拖拽）
- 格式：

  StopDrag()
- 支持端口：29999
- 返回：

  ErrorID,{},StopDrag();
- 示例：

  StopDrag()//退出拖拽

不管机器人是否处于关节拖动状态还是力控拖动状态，该指令均退出拖拽模式

3.50 DragSensivity

- 功能：设置拖拽灵敏度
- 格式：

  DragSensivity(index,value)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	轴号 0代表1到6轴均设置为此灵敏度	是	[0,6]
value	int	轴拖拽灵敏度	是	[1,90]

- 返回：

  ErrorID,{},DragSensivity(1,20);

- 示例：

  DragSensivity(1,20)//设置轴1的拖拽灵敏度为20

3.51 GetDO

- 功能：获取数字输出端口状态
- 格式：

  GetDO(index)

- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	输出端口索引：1~MAX，100~1000	是	[1,MAX] [100,1000]

  参数范围中的MAX 根据控制柜类型不同而不同。 一代控制柜MAX=16,二代控制柜 MAX=24, 小型控制柜MAX为8 ,MG6 也是8 。

- 返回：

  ErrorID,{value},GetDO(1);

- 示例：

GetDO(1)//获取输出端口1的开关状态

3.52 GetAO

- 功能：获取模拟量输出端口状态
- 格式：

  GetAO(index)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	输出端口索引	是	1/2

- 返回：

  ErrorID,{value},GetAO(1);

- 示例：

GetAO(1)//获取模拟端口1的模拟量

3.53 GetToolDO

- 功能：获取末端数字输出端口状态
- 格式：

  GetDO(index)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	输出端口索引	是	[1,MAX]

CR20A的MAX为4，其余的机器MAX为2

- 返回：

ErrorID,{value},GetToolDO(1);

- 示例：

GetToolDO(1)//获取末端输出端口1的状态

3.54 GetDOGroup

- 功能：获取一组数字输出组端口状态
- 格式：

GetDOGroup(index1, ..., indexn)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	输出端口索引	是	[1,MAX] [100,1000]

参数范围中的MAX 根据控制柜类型不同而不同。 一代控制柜MAX=16,二代控制柜 MAX=24, 小型控制柜MAX为8 ,MG6 也是8。

- 返回：

ErrorID,{value1,value2,value3,...},GetDOGroup(index1,index2,index3,...);

- 示例：

GetDOGroup(1,2,3)

3.55 SetTool485

- 功能：设置末端485通讯参数
- 格式：

SetTool485(baudrate)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
baudrate	int	设置的波特率	是		>0

参数名	类型	含义	是否必填	默认值	参数范围
parity	string	设置的校验位 "O":奇校验，"E":偶校验，"N":无校验	否	N	O/E/N
stop	int	设置的停止位	否	1	
identify	int	航插序号1~2	否	1	1/2

- 返回：

  ErrorID,{},SetTool485(115200);

- 示例：

  SetTool485(115200);//设置末端485通讯波特率为115200，无校验位，停止位为1，航插1

3.56 SetSafeWallEnable（队列指令）

- 功能：设置安全墙开关
- 格式：

  SetSafeWallEnable(index,value)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	墙的序号	是	[1,8]
value	int	0：关闭，1：开启	是	0/1

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},SetSafeWallEnable(1,1);

- 示例：

  SetSafeWallEnable(1,1)//开启安全墙1

3.57 SetToolPower

- 功能：设置末端工具供电状态
- 格式：

  SetToolPower(status)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
status	int	0:关闭，1：开启	是	0/1

- 返回：

  ErrorID,{},SetToolPower(1);

- 示例：

  SetToolPower(1)//开启末端工具电源

3.58 SetToolMode

- 功能：设置末端模式
- 格式：

  SetToolMode(mode, type)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
mode	int	1：485模式，2：采集模式	是		
type	int	采集模式：个位为第一路模式，十位为第二路模式 0：为0-10V电压输入模式 1：为电流采集模式 2：为0-5V电压输入模式	否		>=0
identify	int	航插序号1~2，可选参数，不填则为1	否	1	1/2参数名参数名

- 返回：

  ErrorID,{},SetToolMode(1);//设置末端模式为485模式

  ErrorID,{},SetToolMode(2,0);

- 示例：

  SetToolMode(2,0);//设置末端双路模式为0~10V电压输入模式

- 注：

  若mode 为485模式，则仅填一个参数变量

  若mode 为采集模式，则可填2/3个变量

3.59 SetBackDistance（队列指令）

- 功能：设置碰撞回退距离
- 格式：

  SetBackDistance(distance)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
distance	double	碰撞后回退距离	是	[0,50]

- 返回：ResultID:算法队列ID

ErrorID,{ResultID},SetBackDistance(10);

- 示例：

  SetBackDistance(10);//设置碰撞回退距离为10

3.60 SetPostCollisionMode（队列指令）

- 功能：设置选择碰撞后是进入停止状态还是暂停状态
- 格式：

  SetPostCollisionMode(mode)

- 支持端口：29999
- 参数详情：

参数名	类型	含义	是否必填	参数范围
mode	int	0, 状态后进入停止状态。 1，碰撞后进入暂停状态	是	0/1

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},SetPostCollisionMode(0);

- 示例：

  SetPostCollisionMode(1);//设置碰撞后进入停止状态

3.61 SetUser

- 功能：设置用户坐标系
- 格式：

  SetUser(index, value[,type])

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	坐标系索引	是	[1,50]
value	string	要设置的坐标值{x,y,z,rx,ry,rz}	是	任意值
type	int	是否使坐标系改动全局生效	否	0或1

- 返回：

  ErrorID,{},SetUser(1,{1,2,3,4,5,6});

  ErrorID,{},SetUser(1,{1,2,3,4,5,6},1);

- 示例：

  SetUser(1,{1,2,3,4,5,6})//设置用户坐标系1的值为{1,2,3,4,5,6}

  SetUser(1,{1,2,3,4,5,6},1)//设置用户坐标系1的值为{1,2,3,4,5,6}，并全局生效

type 的字段为1 之后，本次的坐标系更新将会被控制器保存，后续即便切换退出TCP模式也会生效。

3.62 SetTool

- 功能：设置工具坐标系
- 格式：

  SetTool(index, value[,type])
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	坐标系索引	是	[1,50]
value	string	要设置的坐标值{x,y,z,rx,ry,rz}	是	任意值
type	int	是否使坐标系改动全局生效	否	0或1

- 返回：

  ErrorID,{},SetTool(1,{1,2,3,4,5,6})

  ErrorID,{},SetTool(1,{1,2,3,4,5,6},1)
- 示例：

  SetTool(1,{1,2,3,4,5,6})//设置工具坐标系1的值为{1,2,3,4,5,6}

  SetTool(1,{1,2,3,4,5,6},1)//设置工具坐标系1的值为{1,2,3,4,5,6},并且全局保存

3.63 CalcUser

- 功能：计算用户坐标系
- 格式：

  CalcUser(index, matrix,offset)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	坐标系索引	是	[0,50]
matrix	int	计算的方向 1：index对应的用户坐标系左乘{x,y,z,rx,ry,rz} 0：index对应的用户坐标系右乘{x,y,z,rx,ry,rz}	是	0/1
offset	string	偏移的坐标值 {offsetx,offsety,offsetz,offsetrx,offsetry,offsetrz}	是	任意值

- 返回：

  ErrorID,{x,y,z,rx,ry,rz},CalcUser(1,0,{1,2,3,4,5,6});
- 示例：

CalcUser(1,0,{1,2,3,4,5,6})//计算用户坐标系值为坐标系1左乘{1,2,3,4,5,6}

3.64 CalcTool

- 功能：计算工具坐标系
- 格式：

  CalcTool(index, matrix,offset)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
index	int	坐标系索引	是	[0,50]
matrix	int	计算的方向： 1：index对应的用户坐标系左乘{x,y,z,rx,ry,rz} 0：index对应的用户坐标系右乘{x,y,z,rx,ry,rz}	是	0/1
offset	string	偏移的坐标值 {offsetx,offsety,offsetz,offsetrx,offsetry,offsetrz}	是	任意值

- 返回：

  ErrorID,{x,y,z,rx,ry,rz},CalcTool(1,0,{1,2,3,4,5,6});

- 示例：

  CalcTool(1,0,{1,2,3,4,5,6})//计算工具坐标系值为坐标系1左乘{1,2,3,4,5,6}

3.65 GetInputBool

- 功能：获取输入寄存器bool数值
- 格式：

  GetInputBool(address)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
address	int	地址索引	是	[0,63]

- 返回：

  ErrorID,{value},GetInputBool(address);

- 示例：

  GetInputBool(0)//获取输入寄存器地址位0的bool值

3.66 GetInputInt

- 功能：获取输入寄存器int数值

- 格式：

  GetInputInt(address)

- 支持端口：29999

- 参数详解：

参数名	类型	含义	是否必填	参数范围
address	int	地址索引：0~23	是	[0,23]

- 返回：

  ErrorID,{value},GetInputInt(address);

- 示例：

  GetInputInt(0))//获取输入寄存器地址位0的int值

3.67 GetInputFloat

- 功能：获取输入寄存器float数值

- 格式：

  GetInputFloat(address)

- 支持端口：29999

- 参数详解：

参数名	类型	含义	是否必填	参数范围
address	int	地址索引	是	[0,23]

- 返回：

  ErrorID,{value},GetInputFloat(address);

- 示例：

  GetInputFloat(0))//获取输入寄存器地址位0的float值

3.68 GetOutputBool

- 功能：获取输出寄存器bool数值

- 格式：

  GetOutputBool(address)

- 支持端口：29999

- 参数详解：

参数名	类型	含义	是否必填	参数范围
address	int	地址索引	是	[0,63]

- 返回：

ErrorID,{value},GetOutputBool(address);

- 示例：

  GetOutputBool(0))//获取输出寄存器地址位0的bool值

3.69 GetOutputInt

- 功能：获取输出寄存器int数值
- 格式：

  GetOutputInt(address)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
address	int	地址索引	是	[0,23]

- 返回：

  ErrorID,{value},GetOutputInt(address);

- 示例：

  GetOutputInt(0))//获取输出寄存器地址位0的int值

3.70 GetOutputFloat

- 功能：获取输出寄存器float数值
- 格式：

  GetOutputFloat(address)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
address	int	地址索引	是	[0,23]

- 返回：

  ErrorID,{value},GetOutputFloat(address);

- 示例：

  GetOutputFloat(0)//获取输出寄存器地址位0的float值

3.71 SetOutputBool

- 功能：设置输出寄存器bool数值
- 格式：

  SetOutputBool(address, value)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
address	int	地址索引	是	[0,63]
value	int	设置值	是	0/1

- 返回：

  ErrorID,{},SetOutputBool(address, value);

- 示例：

  SetOutputBool(0, 1)//设置输出寄存器地址位0的bool值为1

3.72 SetOutputInt

- 功能：设置输出寄存器int数值
- 格式：

  SetOutputInt(address, value)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
address	int	地址索引	是	[0,23]
value	int	设置值	是	带符号32位整型

- 返回：

  ErrorID,{},SetOutputInt(address, value);

- 示例：

  SetOutputInt(0, -1)//设置输出寄存器地址位0的bool值为-1

3.73 SetOutputFloat

- 功能：设置输出寄存器float数值
- 格式：

  SetOutputFloat(address, value)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
address	int	地址索引	是	[0,23]
value	float	设置值	是	单精度浮点

- 返回：

ErrorID,{},SetOutputFloat(address, value);

- 示例：

  SetOutputFloat(0, 1.23)//设置输出寄存器地址位0的float值为1.23

3.74 MovJ(队列指令)

- 功能：点到点运动，目标点位为笛卡尔点位。
- 格式：

  MovJ(joint = {j1, j2, j3, j4, j5, j6},user = 1, tool = 0, a = 20, v = 50, cp = 100)

  MovJ(pose= {x,y,z,rx,ry,rz},user = 1, tool = 0, a = 20, v = 50, cp = 100)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6} OR pose= {x,y,z,rx,ry,rz}	string	目标点的坐标值	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 × 10^(308), 1.7976931348623157 × 10^(308))
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50]int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50]int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值

- 返回：

  ErrorID,{ResultID},MovJ(pose= {x,y,z,rx,ry,rz},user = 1, tool = 0, a = 20, v = 50, cp = 100);

  ResultID:算法队列ID

- 示例：

  MovJ(joint = {1, 2, 3, 4, 5, 6},user = 1, tool = 0, a = 20, v = 50, cp = 100)

  返回：

  ErrorID,{1},MovJ(joint = {1, 2, 3, 4, 5, 6},user = 1, tool = 0, a = 20, v = 50, cp = 100);

注：关节变量进行关节运动，user/tool参数设置无效

3.75 MovL(队列指令)

- 功能：直线运动，目标点位为笛卡尔点位。
- 格式：

  MovL(joint = {j1, j2, j3, j4, j5, j6},user = 1, tool = 0, a = 20, v = 50, cp = 100)

  MovL(pose= {x,y,z,rx,ry,rz},user = 1, tool = 0, a = 20, v = 50, cp = 100)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6} OR pose= {x,y,z,rx,ry,rz}	string	点的坐标值	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 × 10^(308), 1.7976931348623157 × 10^(308))
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
speed=1000	string	速度数值	否		>0 int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值
r=20	string	过渡半径，单位mm	否		[0,100] int值

- 返回：

  ErrorID,{ResultID},MovL(pose= {x,y,z,rx,ry,rz},user = 1, tool = 0, a = 20, v = 50, cp = 100) ;

  ResultID:算法队列ID
- 示例：

  MovL(pose= {x,y,z,rx,ry,rz},user = 1, tool = 0, a = 20, v = 50, cp = 100)

  返回：

  ErrorID,{2},MovL(pose= {x,y,z,rx,ry,rz},user = 1, tool = 0, a = 20, v = 50, cp = 100) ;

  注：

  运动指令参数cp 和r 同时存在，以r为优先

运动指令参数v 和speed 同时存在，以和speed为优先

3.76 MovLIO(队列指令)

- 功能：在直线运动时并行设置数字输出端口状态，目标点位为笛卡尔点位。
- 格式：

  MovLIO(joint = {j1, j2, j3, j4, j5, j6},{Mode,Distance,Index,Status},...,{Mode,Distance,Index,Status},user = 1, tool = 0, a = 20, v = 50, cp = 100)

  MovLIO(pose= {x,y,z,rx,ry,rz},{Mode,Distance,Index,Status},...,{Mode,Distance,Index,Status},user = 1, tool = 0, a = 20, v = 50, cp = 100)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6}|pose= {x,y,z,rx,ry,rz}	string	目标点的坐标值	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 × 10^(308), 1.7976931348623157 × 10^(308))
{Mode,Distance,Index,Status}...	string	可设置多组，最少一组数据	是		详见下表
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指	[1,100]int值

参数名	类型	含义	是否必填	默认值	参数范围
				令设置值	
speed=1000	string	速度数值	否		>0 int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值
r=20	string	过渡半径，单位mm	否		>0 int值

-

{Mode,Distance,Index,Status}	类型	含义	是否必填	参数范围
Mode	int	设置Distance模式 0：Distance为距离百分比; 1：Distance为离起始点或目标点的距离	是	0/1
Distance	int	运行指定的距离: 若Mode为0，则Distance表示起始点与目标点之间距离的百分比 若Distance取值为正，则表示离起始点的距离; 若Distance取值为负，则表示离目标点的距离	是	[0,100] OR 任意值
Index	int	数字输出索引	是	[1,16] [100,1000]
Status	int	数字输出状态	是	0/1

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},MovLIO(pose= {x,y,z,rx,ry,rz},{Mode,Distance,Index,Status},...,{Mode,Distance,Index,Status},user = 1, tool = 0, a = 20, v = 50, cp = 100);

- 示例：

  MovLIO(joint= {1,2,3,4,56},{0,50,1,0})

  注：运动指令参数cp 和r 同时存在，以r为优先

  运动指令参数v 和speed 同时存在，以和speed为优先

  若Mode为0，Distance不在[0,100]范围内，则报参数超限错误

3.77 MovJIO(队列指令)

- 功能：点到点运动时并行设置数字输出端口状态，目标点位为笛卡尔点位。

- 格式：
- MovJIO(joint = {j1, j2, j3, j4, j5, j6},{Mode,Distance,Index,Status},..., {Mode,Distance,Index,Status},user = 1, tool = 0, a = 20, v = 50, cp = 100)
- MovJIO(pose= {x,y,z,rx,ry,rz},{Mode,Distance,Index,Status},...,{Mode,Distance,Index,Status},user = 1, tool = 0, a = 20, v = 50, cp = 100)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6}|pose= {x,y,z,rx,ry,rz}	string	目标点的坐标值	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 × 10^(308), 1.7976931348623157 × 10^(308))
{Mode,Distance,Index,Status}...	string	可设置多组，最少一组数据	是		详见下表
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值

-

{Mode,Distance,Index,Status}	类型	含义	是否必填	参数范围
Mode	int	设置Distance模式 0：Distance为距离百分比； 1：Distance为离起始点或目标点的距离	是	0/1
Distance	int	运行指定的距离: 若Mode为0，则Distance表示起始点与目标点之间距离的百分比 若Distance取值为正，则表示离起始点的距离; 若Distance取值为负，则表示离目标点的距离	是	[0,100] OR 任意值
Index	int	数字输出索引	是	[1,MAX] [100,1000]
Status	int	数字输出状态	是	0/1

index的参数范围中的MAX 根据控制柜类型不同而不同。 一代控制柜MAX=16,二代控制柜 MAX=24, 小型控制柜MAX为8 ,MG6 也是8。

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},MovJIO(pose= {x,y,z,rx,ry,rz},{Mode,Distance,Index,Status},..., {Mode,Distance,Index,Status},user = 1, tool = 0, a = 20, v = 50, cp = 100);

- 示例：

  MovJIO(joint= {1,2,3,4,5,6},{0,50,1,0})

3.78 Arc(队列指令)

- 功能：：从当前位置以圆弧插补方式移动至笛卡尔坐标系下的目标位置。

  该指令需结合其他运动指令确定圆弧起始点。

- 格式：

  Arc(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6},user = 1, tool = 0, a = 20, v = 50, cp = 100, mode=1)

  Arc(pose= {x,y,z,rx,ry,rz},pose= {x,y,z,rx,ry,rz},user = 1, tool = 0, a = 20, v = 50, cp = 100, mode=1)

- 支持端口：29999

- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6} OR pose= {x,y,z,rx,ry,rz}	string	表示圆弧中间点坐标值	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 × 10^(308), 1.7976931348623157 × 10^(308))

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6} OR pose={x,y,z,rx,ry,rz}	string	表示圆弧目标点坐标值	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：[-180,180]
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
speed=1000	string	速度数值	否		>0 int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值
r=20	string	过渡半径，单位mm	否		>0 int值
mode	int	0表示起始按姿态按Slerp插值，目标点姿态可达； 1表示起始姿态按圆弧Z轴旋转，目标点姿态不可达	否	0	0/1

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},Arc(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6},user = 1, tool = 0, a = 20, v = 50, cp = 100, mode=1);

- 示例：

  Arc(joint = {1, 2, 3, 4, 5, 6},joint = {7, 8, 9, 10, 11, 12},user = 1, tool = 0, a = 20, v = 50, cp = 100, mode=1)

  注：

  运动指令参数cp 和r 同时存在，以r为优先

  运动指令参数v 和speed 同时存在，以和speed为优先

3.135 ArcIO(队列指令)

- 功能：在圆弧运动时并行设置数字输出端口状态，目标点位为笛卡尔点位。
- 格式：

ArcIO(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6}, {Mode,Distance,Index,Status},...,{Mode,Distance,Index,Status},user = 1, tool = 0, a = 20, v = 50, cp = 100, mode=1)

ArcIO(pose= {x,y,z,rx,ry,rz},pose= {x,y,z,rx,ry,rz},{Mode,Distance,Index,Status},..., {Mode,Distance,Index,Status},user = 1, tool = 0, a = 20, v = 50, cp = 100, mode=1)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6} OR pose= {x,y,z,rx,ry,rz}	string	表示圆弧中间点坐标值	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 × 10^(308), 1.7976931348623157 × 10^(308))
joint = {j1, j2, j3, j4, j5, j6} OR pose= {x,y,z,rx,ry,rz}	string	表示圆弧目标点坐标值	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：[-180,180]
{Mode,Distance,Index,Status}...	string	可设置多组，最少一组数据	是		详见下表
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
speed=1000	string	速度数值	否		>0 int值
cp=100	string	过渡百分比	否	上一次全局指	[0,100]int值

参数名	类型	含义	是否必填	默认值	参数范围
				令设置值	
r=20	string	过渡半径，单位mm	否		>0 int值
mode=1	int	运动模式	否		[0,2] 0:线性模式 1:过中间点模式 2:固定模式

-

{Mode,Distance,Index,Status}	类型	含义	是否必填	参数范围
Mode	int	设置Distance模式 0：Distance为距离百分比； 1：Distance为离起始点或目标点的距离	是	0/1
Distance	int	运行指定的距离: 若Mode为0，则Distance表示起始点与目标点之间距离的百分比 若Distance取值为正，则表示离起始点的距离; 若Distance取值为负，则表示离目标点的距离	是	[0,100] OR 任意值
Index	int	数字输出索引	是	[1,16] [100,1000]
Status	int	数字输出状态	是	0/1

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},ArcIO(pose= {x,y,z,rx,ry,rz},pose= {x,y,z,rx,ry,rz}, {Mode,Distance,Index,Status},...,{Mode,Distance,Index,Status},user = 1, tool = 0, a = 20, v = 50, cp = 100);

- 示例：

  ArcIO(pose={-1140.580322,-31.398853,93.642189,10.629999,21.659998,-86.040001},pose={-1220.207031,-281.265533,93.642189,10.629999,21.659998,-86.040001},{0,25,1,1},{0,50,2,1},{0,75,3,1},{0,100,4,1},user=1,tool=2,a=20,v=50,cp=100)

  注：运动指令参数cp 和r 同时存在，以r为优先

  运动指令参数v 和speed 同时存在，以和speed为优先

  若Mode为0，Distance不在[0,100]范围内，则报参数超限错误

3.79 Circle(队列指令)

- 功能：：整圆运动
- 格式：

Circle(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6},1,user = 1, tool = 0, a = 20, v = 50, cp = 100, mode=1)

Circle(pose= {x,y,z,rx,ry,rz},pose= {x,y,z,rx,ry,rz},1,user = 1, tool = 0, a = 20, v = 50, cp = 100, mode=1)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6} | pose= {x,y,z,rx,ry,rz}	string	表示圆弧中间点坐标值	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 × 10^(308), 1.7976931348623157 × 10^(308))
joint = {j1, j2, j3, j4, j5, j6} | pose= {x,y,z,rx,ry,rz}	string	表示圆弧目标点坐标值	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：[-180,180]
count	int	整圆个数	是		>0
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
speed=1000	string	速度数值	否		>0 int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值
r=20	string	过渡半径，单位mm	否		>0 int值
mode=1	int	运动模式	否		[0,2] 0:线性模式 1:过中间点模式 2:固定模式

返回：

ErrorID,{ResultID},Circle(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6},1，user = 1, tool = 0, a = 20, v = 50, cp = 100, mode=1);

ResultID:算法队列ID

- 示例：

  Circle(joint = {1, 2, 3, 4, 5, 6},joint = {7, 8, 9, 10, 11, 12},1,user = 1, tool = 0, a = 20, v = 50, cp = 100, mode=1);

  注：运动指令参数cp 和r 同时存在，以r为优先

  运动指令参数v 和speed 同时存在，以和speed为优先

3.80 MoveJog

- 功能：点动运动，不固定距离运动
- 格式：

  MoveJog(axisID,coordtype=typeValue,user=index,tool=index)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
axisID	string	点动运动轴 J1+ 表示关节1正方向运动 J1- 表示关节1负方向运动 J2+ 表示关节2正方向运动 J2- 表示关节2负方向运动 J3+ 表示关节3正方向运动 J3- 表示关节3负方向运动 J4+ 表示关节4正方向运动 J4- 表示关节4负方向运动 J5+ 表示关节5正方向运动 J5- 表示关节5负方向运动 J6+ 表示关节6正方向运动 J6- 表示关节6负方向运动 X+ 表示X轴正方向运动 X- 表示X轴负方向运动 Y+ 表示Y轴正方向运动 Y- 表示Y轴负方向运动 Z+ 表示Z轴正方向运动 Z- 表示Z轴负方向运动 Rx+ 表示Rx轴正方向运动 Rx- 表示Rx轴负方向运动 Ry+ 表示Ry轴正方向运动 Ry- 表示Ry轴负方向运动 Rz+ 表示Rz轴正方向运动 Rz- 表示Rz轴负方向运动	是		指定字符串值
coordtype=0	string	0:关节点动 1:用户坐标系 2:工具坐标系	否	上一次全局指令设置值	0/1/2
user=1	string	用户索引	否	上一次全局指令设置值	[0,50]
tool=1	string	工具索引	否	上一次全局指令设置值	[0,50]

- 返回：

  ErrorID,{},MoveJog(axisID,coordtype=typeValue,user=index,tool=index);

- 示例（1）：

  MoveJog(J2-) //J2负方向运动，再停止点动

- 返回：

  0,{},MoveJog(J2-);

- 示例（2）：

  MoveJog() //停止运动

  示例（3）：

  MoveJog(X+,coordtype=0) //J1轴关节运动

- 返回：

  0,{},MoveJog();

- 说明：

  其中用户若是在发关节点动运行则会忽略coordtype、User以及Tool这三个可选设置参数，执行关节运动；

  用户若发送X+/Y+/Z+/Rx+/Ry+/Rz+/X-/Y-/Z-/Rx-/Ry-/Rz-，则会根据coordtype类型指定运动类型，当coordtype不为1/2时则指令会报错-6，运动轴与运动类型不匹配

  命令下发后，须另外下发MoveJog()停止命令控制机器人停止运动；另下发非指定string内容的参数都会导致机器人停止；

3.81 StartPath

- 功能：轨迹复现。

  StartPath(traceName,isConst=0,multi=1,sample=50,freq=0.2,user=0,tool=0)

- 支持端口：29999

- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
traceName	string	轨迹文件名（.csv） 轨迹路径存放在/dobot/userdata/project/process/trajectory/	是		.csv文件
isConst=0	string	1:为匀速复现； 0：非匀速复现	否	0	0/1
multi=1	string	复现倍率	否	1	[0.25,2]
sample=50	string	复现间隔，单位 ms	否	50	[8,1000]
freq=0.2	string	滤波系数，值越小表示拟合后曲线越平滑，但轨迹变形越严重。当取值为1时，表示关闭滤波。请根据输入点位的平滑程度选择适当的滤波系数。	否	0.2	(0,1]

参数名	类型	含义	是否必填	默认值	参数范围
user=0	string	用户坐标系索引	否	上一次全局指令设置值	[0,50]
tool=0	string	工具坐标系索引	否	上一次全局指令设置值	[0,50]

- 返回：

  ErrorID,{},StartPath(traceName);

- 示例：

  StartPath(recv_string.csv)

  返回：

  0,{},StartPath(recv_string.csv);

- 注：

  需将发送端的编码方式选择为UTF-8格式，否则会导致中文字符接收异常

  当前轨迹复现指令不会将轴走到首点，需调用获取首点指令后将轴运动到首点，再进行轨迹复现。

  用户可以通过获取RobotMode查询机器人运行状态

  若在ROBOT_MODE_RUNNING表示机器人在轨迹拟合运行中，达到ROBOT_MODE_ENABLE表示轨迹拟合运行完成，ROBOT_MODE_ERROR表示报警；

3.82 GetStartPose

- 功能： 获取轨迹的第一个点位
- 格式：

  GetStartPose(traceNamel,pathType)

- 支持端口：29999

- 参数详解：

参数名	类型	含义	是否必填	参数范围
traceName	string	轨迹文件名（.csv） 轨迹路径存放 在/dobot/userdata/project/process/trajectory/ 或 /dobot/userdata/project/process/track/	是	.csv文件
pathType	int	轨迹的类型 1: 默认，用于复现的轨迹，轨迹文件名（.csv） 轨迹路径存放 在/dobot/userdata/project/process/trajectory/ 2: 用于拟合的轨迹，轨迹文件名（.csv） 轨迹路径存放 在/dobot/userdata/project/process/track/	否	不填 或者 1 或者 2

- 示例：

  GetStartPose(recv_string.csv)//获取名字recv_string的轨迹的首个点的对应信息

- 返回：

  ErrorID,{pointtype,{j1,j2,j3,j4,j5,j6},user,tool,{x,y,z,rx,ry,rz}},GetStartPose(traceNamel);

  {pointtype}:返回点位类型，0:示教常量；1：关节变量；2：位姿变量

- 示例：

  ErrorID,{0,{10,20,20,20,20,20},1,1,{x,y,z,rx,ry,rz}},GetStartPose(“test.csv”);

  ErrorID,{1,{j1,j2,j3,j4,j5,j6}},GetStartPose(traceNamel);

  ErrorID,{2,{x,y,z,rx,ry,rz}},GetStartPose(traceNamel);

  ErrorID,{2,{x,y,z,rx,ry,rz}},GetStartPose(traceNamel,2); **

  ErrorID,{1,{j1,j2,j3,j4,j5,j6}},GetStartPose(traceNamel,2);

3.83 RelMovJTool(队列指令)

- 功能：沿工具坐标系进行相对运动指令，末端运动方式为关节运动。
- 格式：

  RelMovJTool(x, y, z, rx, ry, rz, user = 1, tool = 0, a = 20, v = 50, cp = 100)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
x	double	X轴方向偏移，单位：mm	是		任意值
y	double	Y轴方向偏移，单位：mm	是		任意值
z	double	Z轴方向偏移，单位：mm	是		任意值
rx	double	Rx 轴位置，单位：度	是		任意值
ry	double	Ry 轴位置，单位：度	是		任意值

参数名	类型	含义	是否必填	默认值	参数范围
rz	double	Rz 轴位置，单位：度	是		任意值
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值

- 返回：

  ErrorID,{ResultID},RelMovJTool(x, y, z, rx, ry, rz, user = 1, tool = 0, a = 20, v = 50, cp = 100);

  {ResultID}:算法队列ID

- 示例：

  RelMovJTool(10,10,10,0,0,0)

3.84 RelMovLTool(队列指令)

- 功能：沿工具坐标系进行相对运动指令，末端运动方式为直线运动。
- 格式：

  RelMovLTool(x, y, z, rx, ry, rz, user = 1, tool = 0, a = 20, v = 50, cp = 100)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
x	double	X轴方向偏移，单位：mm	是		任意值
y	double	Y轴方向偏移，单位：mm	是		任意值
z	double	Z轴方向偏移，单位：mm	是		任意值
rx	double	Rx 轴位置，单位：度	是		任意值
ry	double	Ry 轴位置，单位：度	是		任意值
rz	double	Rz 轴位置，单位：度	是		任意值
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值

参数名	类型	含义	是否必填	默认值	参数范围
speed=1000	string	速度数值	否		>0 int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值
r=20	string	过渡半径，单位mm	否		>0 int值

- 返回：

  ErrorID,{ResultID},RelMovLTool(x, y, z, rx, ry, rz, user = 1, tool = 0, a = 20, v = 50, cp = 100);

  {ResultID}:算法队列ID

- 示例：

  RelMovLTool(10,10,10,0,0,0)

3.85 RelMovJUser(队列指令)

- 功能：沿用户坐标系进行相对运动指令，末端运动方式为关节运动。
- 格式：

  RelMovJUser(x, y, z, rx, ry, rz, user = 1, tool = 0, a = 20, v = 50, cp = 100)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
x	double	X轴方向偏移，单位：mm	是		任意值
y	double	Y轴方向偏移，单位：mm	是		任意值
z	double	Z轴方向偏移，单位：mm	是		任意值
rx	double	Rx 轴位置，单位：度	是		任意值
ry	double	Ry 轴位置，单位：度	是		任意值
rz	double	Rz 轴位置，单位：度	是		任意值
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值

- 返回：

  ErrorID,{ResultID},RelMovJUser(x,y,z,rx,ry,rz,user = 1, tool = 0, a = 20, v = 50, cp = 100);

  {ResultID}:算法队列ID

- 示例：

  RelMovJUser(10,10,10,0,0,0)

3.86 RelMovLUser(队列指令)

- 功能：沿用户坐标系进行相对运动指令，末端运动方式为直线运动。
- 格式：

  RelMovLUser(x, y, z, rx, ry, rz, user = 1, tool = 0, a = 20, v = 50, cp = 100)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
x	double	X轴方向偏移，单位：mm	是		任意值
y	double	Y轴方向偏移，单位：mm	是		任意值
z	double	Z轴方向偏移，单位：mm	是		任意值
rx	double	Rx 轴位置，单位：度	是		任意值
ry	double	Ry 轴位置，单位：度	是		任意值
rz	double	Rz 轴位置，单位：度	是		任意值
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
speed=1000	string	速度数值	否		>0 int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值

- 返回：

  ErrorID,{ResultID},RelMovLUser(x,y,z,rx,ry,rz,user = 1, tool = 0, a = 20, v = 50, cp = 100);

  {ResultID}:算法队列ID
- 示例：

  RelMovLUser(10,10,10,0,0,0)

  注：

  运动指令参数cp 和r 同时存在，以r为优先

  运动指令参数v 和speed 同时存在，以和speed为优先

3.87 RelJointMovJ(队列指令)

- 功能：沿各轴关节坐标系进行相对运动指令，末端运动方式为关节运动。
- 格式：

  RelJointMovJ(Offset1, Offset2, Offset3, Offset4, Offset5, Offset6)

  RelJointMovJ(Offset1, Offset2, Offset3, Offset4, Offset5, Offset6, user = 1, tool = 0, a = 20, v = 50, cp = 100)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
Offset1	double	关节1的偏移值，单位：度	是		任意值
Offset2	double	关节2的偏移值，单位：度	是		任意值
Offset3	double	关节3的偏移值，单位：度	是		任意值
Offset4	double	关节4的偏移值，单位：度	是		任意值
Offset5	double	关节5的偏移值，单位：度	是		任意值
Offset6	double	关节6的偏移值，单位：度	是		任意值
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值

- 返回：

  ErrorID,{ResultID},RelJointMovJ(Offset1,Offset2,Offset3,Offset4,Offset5,Offset6,user = 1, tool = 0, a = 20, v = 50, cp = 100);

  {ResultID}:算法队列ID
- 示例：

  RelJointMovJ(10,10,10,0,0,0)

3.88 GetCurrentCommandID()

- 功能：返回当前算法运行队列ID
- 格式：

  GetCurrentCommandID()
- 支持端口：29999
- 返回：

  ErrorID,{ResultID},GetCurrentCommandID();
- 示例：

  0,{100},GetCurrentCommandID();

- 示例：队列指令怎么判断执行完毕?

  队列指令为立即返回指令，接口返回成功仅代表发送成功，不代表执行完毕。若判断执行完毕，则需要结合CommandID和RobotMode来综合判断

```
MovJ(P1)
uint64_t p2Id = parseResultId(MovJ(P2));
MovJ(P3)

while(true) {
  uint64_t currentId = parseResultId (GetCurrentCommndID());
  if (currentId > p2Id) { // currentId执行到P2后面的点。
    break; // 退出等待，P2点执行完成
  }
  Sleep(1);
}
```

```
MovJ(P1)
uint64_t p2Id = parseResultId(MovJ(P2));

while(true) {
  uint64_t currentId = parseResultId (GetCurrentCommndID());
  bool isStop = parseResultId (RobotMode ()) == 5; // ROBOT_MODE_ENABLE
  if (currentId == p2Id && isStop ) { // currentId等于p2Id时P2点并执行完成。
    break; // 退出等待，P2点执行完成
  }
  Sleep(1);
}
```

3.89 EnableSafeSkin()(队列指令)

- 功能：启动/停止安全皮肤功能
- 格式：

  EnableSafeSkin(status)

- 支持端口：29999
- 参数详解：
参数名	类型	含义	是否必填	参数范围
status	int	0代表关闭，1代表开启	是	0/1

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},EnableSafeSkin();

- 示例：

  EnableSafeSkin(1);

  注：ErrorID为-1 可能的情况是当前无电子皮肤

3.90 SetSafeSkin()(队列指令)

- 功能：设置安全皮肤的灵敏度
- 格式：

SetSafeSkin(part, status)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
part	int	皮肤部件 arm（小臂安全皮肤）=3、J4=4、J5=5、J6=6	是	3/4/5/6
status	int	灵敏度阈值 0表示关闭，1表示low,2表示middle，3表示high	是	0/1/2/3

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},SetSafeSkin(part, status);

- 示例：

  格式：SetSafeSkin(3, 1)；

3.91 ServoJ()(队列指令)

- 功能：基于关节空间的动态跟随命令。
- 格式：

  ServoJ(J1,J2,J3,J4,J5,J6, t=0.1, aheadtime=50, gain=500)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	参数范围
J1	double	点J1 轴位置，单位：度	是	
J2	double	点J2 轴位置，单位：度	是	
J3	double	点J3 轴位置，单位：度	是	
J4	double	点J4 轴位置，单位：度	是	
J5	double	点J5 轴位置，单位：度	是	
J6	double	点J6 轴位置，单位：度	是	
t	float	该点位的运行时间，默认0.1,单位：s	否	[0.004,3600.0]
aheadtime	float	作用类似于PID的D项，默认50，标量，无单位	否	[20.0,100.0]
gain	float	目标位置的比例放大器，作用类似于PID的P项，默认500，标量，无单位	否	[200.0,1000.0]

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},ServoJ(J1,J2,J3,J4,J5,J6, t=0.1, aheadtime=50, gain=500);

- 示例：

  格式：ServoJ(0,0,-90,0,90,0)

3.92 ServoP()(队列指令)

功能：基于笛卡尔空间的动态跟随命令。

格式：

ServoP(X,Y,Z,Rx,Ry,Rz, t=0.1, aheadtime=50, gain=500)

支持端口：29999

参数详解：

参数名	类型	含义	是否必填	参数范围
X	double	X 轴位置，单位：毫米	是	
Y	double	Y 轴位置，单位：毫米	是	
Z	double	Z 轴位置，单位：毫米	是	
Rx	double	Rx 轴位置，单位：度	是	
Ry	double	Ry 轴位置，单位：度	是	
Rz	double	Rz 轴位置，单位：度	是	
t	float	该点位的运行时间，默认0.1,单位：s	否	[0.004,3600.0]
aheadtime	float	作用类似于PID的D项，默认50，标量，无单位	否	[20.0,100.0]
gain	float	目标位置的比例放大器，作用类似于PID的P项，默认500，标量，无单位	否	[200.0,1000.0]

返回：ResultID:算法队列ID

ErrorID,{ResultID},ServoP(X,Y,Z,Rx,Ry,Rz, t=0.1, aheadtime=50, gain=500);

示例：

格式：ServoP(-500,100,200,150,0,90)

3.93 CheckOddMovL()

- 功能：检查两个点位直线运动过程中点位可达性
- 命令：仅支持在静止状态下进行查询，点位类型为关节点动
- 格式：

  CheckOddMovL(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6}, user = 1, tool = 0, a = 20, v = 50, cp = 100)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6}	string	起点的坐标值	是		joint值：任意值

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6}	string	目标点的坐标值	是		joint值：任意值
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
speed=1000	string	速度数值	否		>0 int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值
r=20	string	过渡半径，单位mm	否		>0 int值

返回：**ResultID: 校验返回值**

ErrorID,{ResultID},CheckOddMovL(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6},user = 1, tool = 0, a = 20, v = 50, cp = 100)

ResultID：

- 校验通过：0
- 无法校验：-1
- 校验错误：报警码见报警id(id含义和算法错误码一致，但不会进行报错)

注：若可选参数指定用户坐标系/工具坐标系，则对两个点位都生效

参数v 和speed 同时存在，以和speed为优先

3.94 CheckOddMovJ()

- 功能：检查两个点位关节运动过程中点位可达性
- 命令：仅支持在静止状态下进行查询，点位类型为关节点动
- 格式：

CheckMovJ(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6})

CheckMovJ(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6}, user = 1, tool = 0, a = 20, v = 50, cp = 100)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6}	string	起点的坐标值	是		joint值：任意值

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6}	string	目标点的坐标值	是		joint值：任意值
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
speed=1000	string	速度数值	否		>0 int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值

返回：**ResultID: 校验返回值**

ErrorID,{ResultID},CheckMovJ(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6},user = 1, tool = 0, a = 20, v = 50, cp = 100)

ResultID：

- 校验通过：0
- 无法校验：-1
- 校验错误：报警码见报警id(id含义和算法错误码一致，但不会进行报错)

注：若可选参数指定用户坐标系/工具坐标系，则对两个点位都生效

参数v 和speed 同时存在，以和speed为优先

3.95 CheckOddMovC()

- 功能：检查经过指定三个点位的圆弧运动过程中点位可达性
- 命令：仅支持在静止状态下进行查询，点位类型为关节点动
- 格式：

CheckMovC(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6})

CheckMovC(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6}, {user = 1, tool = 0, a = 20, v = 50, cp = 100})

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6}	string	起点的坐标值	是		joint值：任意值
joint = {j1, j2, j3, j4, j5, j6}	string	圆弧起点的坐标值	是		joint值：任意值

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6}	string	圆弧终点的坐标值	是		joint值：任意值
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
speed=1000	string	速度数值	否		>0 int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值
r=20	string	过渡半径，单位mm	否		>0 int值

返回：**ResultID: 校验返回值**

ErrorID,{ResultID},CheckMovC(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6},user = 1, tool = 0, a = 20, v = 50, cp = 100)

ResultID：

- 校验通过：0
- 无法校验：-1
- 校验错误：报警码见报警id(id含义和算法错误码一致，但不会进行报错)

注：若可选参数指定用户坐标系/工具坐标系，则对两个点位都生效

参数v 和speed 同时存在，以和speed为优先

3.96 LogExportUSB()

- 功能：控制器内部日志导出
- 格式：

  LogExportUSB(0)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填
index	int	导出格式： 0：导出logs/all + logs/user 1：导出logs文件夹全部内容	是

返回：

ErrorID,{},LogExportUSB(0);

注：该接口为非阻塞指令，发送成功后即返回，若查询具体导出结果，请使用GetExportStatus()

若在导出过程中下发导出指令，则会返回-1，指令执行失败

3.97 GetExportStatus()

- 功能：获取日志导出状态
- 格式：

  GetExportStatus()
- 支持端口：29999
- 返回值：

返回值	描述
0	未开始拷贝
1	拷贝中
2	拷贝完成
3	拷贝失败,未插入U盘
4	拷贝失败,U盘空间不足
5	拷贝失败,拷贝过程中拔出u盘导致失败

返回：

ErrorID,{index},GetExportStatus();

注：导出状态会一直保持，直到下次使用导出功能

3.98 RelPointUser()

- 功能：沿用户坐标系笛卡尔点偏移
- 格式：

  RelPointUser(joint = {j1, j2, j3, j4, j5, j6}, {offsetX,offsetY,offsetZ,offsetRx,offsetRy,offsetRz})

  RelPointUser(pose= {x,y,z,rx,ry,rz}, {offsetX,offsetY,offsetZ,offsetRx,offsetRy,offsetRz})
- 支持端口：29999

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6} | pose= {x,y,z,rx,ry,rz}	string	表示偏移起始点位	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 × 10^(308), 1.7976931348623157 × 10^(308))

参数名	类型	含义	是否必填	默认值	参数范围
{offsetX,offsetY,offsetZ,offsetRx,offsetRy,offsetRz}	double	笛卡尔坐标系下X轴、Y轴、Z轴、Rx轴、Ry轴、Rz轴方向上的偏移	是		任意值

- 返回值：

  ErrorID,{X,Y,Z,Rx,Ry,Rz},RelPointUser();

  {X,Y,Z,Rx,Ry,Rz}表示笛卡尔坐标值；

3.99 RelPointTool()

- 功能：沿工具坐标系笛卡尔点偏移
- 格式：

  RelPointTool(joint = {j1, j2, j3, j4, j5, j6}, {offsetX,offsetY,offsetZ,offsetRx,offsetRy,offsetRz})

  RelPointTool(pose= {x,y,z,rx,ry,rz}, {offsetX,offsetY,offsetZ,offsetRx,offsetRy,offsetRz})

- 支持端口：29999

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6} | pose= {x,y,z,rx,ry,rz}	string	表示偏移起始点位	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 × 10^(308), 1.7976931348623157 × 10^(308))

参数名	类型	含义	是否必填	默认值	参数范围
{offsetX,offsetY,offsetZ,offsetRx,offsetRy,offsetRz}	double	笛卡尔坐标系下X轴、Y轴、Z轴、Rx轴、Ry轴、Rz轴方向上的偏移	是		任意值

- 返回值：

  ErrorID,{X,Y,Z,Rx,Ry,Rz},RelPointTool();

  {X,Y,Z,Rx,Ry,Rz}表示笛卡尔坐标值；

3.100 RelJoint()

- 功能：关节点位偏移
- 格式：

  RelJoint(J1,J2,J3,J4,J5,J6,{offset1,offset2,offset3,offset4,offset5,offset6})

- 支持端口：29999

参数名	类型	含义	是否必填	参数范围
J1	double	点J1 轴位置，单位：度	是	
J2	double	点J2 轴位置，单位：度	是	
J3	double	点J3 轴位置，单位：度	是	
J4	double	点J4 轴位置，单位：度	是	
J5	double	点J5 轴位置，单位：度	是	
J6	double	点J6 轴位置，单位：度	是	
{offset1,offset2,offset3,offset4,offset5,offset6}	double	{关节1/2/3/4/5/6的偏移值}，单位：度	是	

- 返回值：

ErrorID,{J1,J2,J3,J4,J5,J6},RelJoint();

{J1,J2,J3,J4,J5,J6}表示关节值；

3.101 WeldArcSpeedStart()

- 功能：焊接过程运动速度以绝对速度规划的启动指令(队列指令)
- 格式：

  WeldArcSpeedStart()
- 支持端口：29999
- **输入参数：**无
- 返回值：

  ErrorID,{commandID},WeldArcSpeedStart();

  示例:

  发送：

  WeldArcspeedStart()

  返回：

  0,{25},WeldArcspeedStart();

  0：下发成功

  25：入队索引号为25

  备注：请搭配WeldArcSpeed()和WeldArcSpeedEnd()使用，下方给出简单使用demo

3.102 WeldArcSpeedEnd()

- 功能：焊接过程运动速度以绝对速度规划的停止指令(队列指令)
- 格式：

  WeldArcSpeedEnd()
- 支持端口：29999
- 返回值：

  ErrorID,{commandID},WeldArcSpeedEnd();

  示例:

  发送：

  WeldArcSpeedEnd()

  返回：

  0,{29},WeldArcSpeedEnd();

  0：下发成功

29：入队索引号为29

备注：请搭配WeldArcSpeed()和WeldArcSpeedStart()使用，下方给出简单使用demo

3.103 WeldArcSpeed()

- 功能：不受全局速度影响的单位为毫米/秒的只影响焊接运动的速度指令(队列指令)
- 格式：

  WeldArcSpeed(speed)

- 支持端口：29999

参数名	类型	含义	是否必填	默认值	参数范围
speed	int	焊接速度，单位mm/s(参数范围1~20)			

- 返回值：

  ErrorID,{commandID},WeldArcSpeed(speed);

  示例：

  发送：

  WeldArcSpeed(10)

  含义：设置WeldArcSpeedStart()和WeldArcSpeedEnd()之间的movL的 速度为10mm/s

  返回：

  0,{25},weldArcSpeed(10);

  0：下发成功

  25：入队索引号为25

  备注：请搭配WeldArcSpeedStart()和WeldArcSpeedEnd()使用，下方给出简单使用demo

焊接速度设置相关的demo

```
MovL(P1)
WeldArcSpeed(10)
WeldArcSpeedStart()
MovL(P2)
……
WeldArcSpeedEnd()
MovL(P3)

含义：直线运动MovL(P1)按全局速度运动，指定焊接速度，调用WeldArcSpeedStart()开启焊接速度开关后，笛卡尔运动按设置的焊接速度运动，如直线运动MovL(P2)按WeldArcSpeed(10)设置的绝对速度10mm/s的速度运动。调用WeldArcSpeedEnd()关闭焊接速度开关后，直线运动MovL(P3)按全局速度运动。
即：
MovL(P1)：速度为 全局速度 的 直线运动
MovL(P2)：速度为 焊接速度 的 直线运动
MovL(P3)：速度为 全局速度 的 直线运动
备注：wedlArcSpeed/WeldArcSpeedStart/WeldArcSpeedEnd指令只影响规划速度
```

3.104 WeaveStart()

- 功能：焊接过程运动轨迹按照设置参数的启动指令(队列指令)
- 格式：

  WeaveStart()
- 支持端口：29999
- 返回值：

  ErrorID,{commandID},WeaveStart()

  示例:

  发送：

  WeaveStart()

  返回:

  0,{27},WeaveStart();

  0：下发成功

  27：入队索引号为27

  备注：请搭配WeaveParams()和WeaveEnd()使用，下方给出简单使用demo

3.105 WeaveEnd()

- 功能：焊接过程运动轨迹按照设置参数的停止指令(队列指令)
- 格式：

  WeaveEnd()
- 支持端口：29999
- 返回值：

  ErrorID,{commandID},WeaveEnd();

  示例:

  发送：

  WeaveEnd()

  返回：

  0,{29},WeaveEnd();

  0：下发成功

  29：入队索引号为29

  备注：请搭配WeaveParams()和WeaveStart()使用，下方给出简单使用demo

3.106 WeaveParams()

- 功能：设置摆焊的参数

- 格式：

  WeaveParams(weldType, frequency, leftAmplitude, rightAmplitude, direction, stopMode, stopTime1, stopTime2, stopTime3, stopTime4, radius，radian)

- 支持端口：29999

  参数说明：

  必填参数：12

参数名	类型	说明
weldType	int	摆弧类型(三角:1 螺旋:2 梯形:3 正弦:4 月牙:5)
frequency	double	摆焊频率(参数范围0~20)，直线焊为0，非直线焊传入需要大于0的频率
leftAmplitude	double	左摆幅(参数范围0~50)
rightAmplitude	double	右摆幅(参数范围0~50)
Direction	int	0:左启动 1:右启动
stopMode	int	0: 机器人停止 1:摆焊停止
stopTime1	double	三角摆第一个点的停止时间(参数范围0~9.9)
stopTime2	double	三角摆第二个点的停止时间(参数范围0~9.9)
stopTime3	double	三角摆第三个点的停止时间(参数范围0~9.9)
stopTime4	double	三角摆第四个点的停止时间(参数范围0~9.9)
radius	double	螺旋摆焊半径(参数范围0~50)
radian	int	月牙摆弧度 单位% 参数范围(-100~100)

  可选参数：2

参数名	类型	说明
weaveDirectionDO	int	摆弧方向DO输出索引号 根据实际控制柜上的DO数量做参数限制
weavePositionDO	int	摆弧位置DO输出索引号 根据实际控制柜上的DO数量做参数限制

  示例：

  WeaveParams(1,1,2,3,0,0,0,0,0,0,0,0,weaveDirectionDO=1,weavePositionDO=2)

  含义

  摆弧方向DO输出索引号设置为1，摆弧运动下一规划方向是沿焊道向上方向，DO1输出高电平。沿焊道向下方向，DO1输出一个相反的DO信号。

  摆弧位置DO输出索引号设置为2，摆弧运动当前位置位于焊道上方，DO2输出一个高电平信号。位于焊道下方，DO2输出一个相反的DO信号。

  备注：

  1）weaveDirectionDO和weavePositionDO的输入范围为[0,16]，当输入为0或者不带可选参数时，均摆弧过程中不输出DO。

  2）摆弧方向DO输出和摆弧位置DO输出可单独设置其中一个。

  直线

  不需要下发该指令

三角

下发格式：WeaveParams(1, frequency, leftAmplitude, rightAmplitude, direction, stopMode, stopTime1, stopTime2, stopTime3, stopTime4, 0, 0)

参数名	类型	说明
weldType	int	摆弧类型(三角:1 螺旋:2 梯形:3 正弦:4 月牙:5)
frequency	double	摆焊频率(参数范围0~20)，直线焊为0，非直线焊传入需要大于0的频率
leftAmplitude	double	左摆幅(参数范围0~50)
rightAmplitude	double	右摆幅(参数范围0~50)
Direction	int	0:左启动 1:右启动
stopMode	int	0: 机器人停止 1:摆焊停止
stopTime1	double	三角摆第一个点的停止时间(参数范围0~9.9)
stopTime2	double	三角摆第二个点的停止时间(参数范围0~9.9)
stopTime3	double	三角摆第三个点的停止时间(参数范围0~9.9)
stopTime4	double	三角摆第四个点的停止时间(参数范围0~9.9)

螺旋

下发格式：WeaveParams(2, frequency, leftAmplitude, rightAmplitude, 0, 0, 0, 0, 0, 0, radius, 0)

参数名	类型	说明
weldType	int	摆弧类型(三角:1 螺旋:2 梯形:3 正弦:4 月牙:5)
frequency	double	摆焊频率(参数范围0~20)，直线焊为0，非直线焊传入需要大于0的频率
leftAmplitude	double	左摆幅(参数范围0~50)
rightAmplitude	double	右摆幅(参数范围0~50)
radius	double	螺旋摆焊半径(参数范围0~50)

梯形

下发格式：WeldWeaveStart(3, frequency, leftAmplitude, rightAmplitude, direction, 0, 0, 0, 0, 0, 0, 0)

参数名	类型	说明
weldType	int	摆弧类型(三角:1 螺旋:2 梯形:3 正弦:4 月牙:5)
frequency	double	摆焊频率(参数范围0~20)，直线焊为0，非直线焊传入需要大于0的频率
leftAmplitude	double	左摆幅(参数范围0~50)
rightAmplitude	double	右摆幅(参数范围0~50)
Direction	int	0:左启动 1:右启动

正弦

下发格式：WeldWeaveStart(4, frequency, leftAmplitude, rightAmplitude, direction, 0, 0, 0, 0, 0, 0, 0)

参数名	类型	说明
weldType	int	摆弧类型(三角:1 螺旋:2 梯形:3 正弦:4 月牙:5)

参数名	类型	说明
frequency	double	摆焊频率(参数范围0~20)，直线焊为0，非直线焊传入需要大于0的频率
leftAmplitude	double	左摆幅(参数范围0~50)
rightAmplitude	double	右摆幅(参数范围0~50)
Direction	int	0:左启动 1:右启动

月牙

下发格式：WeaveParams(5, frequency, leftAmplitude, rightAmplitude, 0, stopMode, stopTime1, stopTime2, stopTime3, stopTime4, 0, radian)

参数名	类型	说明
weldType	int	摆弧类型(三角:1 螺旋:2 梯形:3 正弦:4 月牙:5)
frequency	double	摆焊频率(参数范围0~20)，直线焊为0，非直线焊传入需要大于0的频率
leftAmplitude	double	左摆幅(参数范围0~50)
rightAmplitude	double	右摆幅(参数范围0~50)
stopMode	int	0: 机器人停止 1:摆焊停止
stopTime1	double	三角摆第一个点的停止时间(参数范围0~9.9)
stopTime2	double	三角摆第二个点的停止时间(参数范围0~9.9)
stopTime3	double	三角摆第三个点的停止时间(参数范围0~9.9)
stopTime4	double	三角摆第四个点的停止时间(参数范围0~9.9)
radian	int	月牙摆弧度 单位% 参数范围(-100~100)

- 返回值：

  ErrorID,{},WeaveParams(params);

  示例：

  发送：

  WeaveParams(1,1,2,3,0,0,0,0,0,0,0)

  返回：

  0,{26},WeaveParams(1,1,2,3,0,0,0,0,0,0,0);

  备注：请搭配WeaveStart()和WeaveEnd()使用，下方给出简单使用demo

摆弧设置相关的demo

```
MovL(P1)
WeldArcSpeed(10)
WeldArcSpeedStart()
MovL(P2)
WeaveParams(1,1,2,3,0,0,0,0,0,0,0)
WeaveStart()
MovL(P3)
Arc(P4, P5)
……
WeaveEnd()
MoveL(P6)
WeldArcSpeedEnd()
MovL(P7)
```

含义：直线运动MovL(P1)按全局速度运动，指定焊接速度，调用WeldArcSpeedStart()开启焊接速度开关后，笛卡尔运动按设置的焊接速度运动，如直线运动MovL(P2)按WeldArcSpeed(10)设置的绝对速度10mm/s的速度运动。
调用WeaveParams(1,1,2,3,0,0,0,0,0,0,0),根据参数表，设置摆弧参数(三角摆类型，摆动频率为1Hz，左摆幅为2mm，右摆幅为3mm，左启动，不作摆弧停留)。调用WeaveStart()开启摆弧开关后，笛卡尔运动都会叠加上摆弧的类型。如直线运动MovL(P3)和Arc(P4,P5)都是焊接速度为10mm/s，按三角摆运动。调用WeaveEnd()关闭摆焊开关后，轨迹恢复非摆弧轨迹。MovL(P6)是以焊接速度为10mm/s的直线运动。
调用WeldArcSpeedEnd()关闭焊接速度开关后，直线运动MovL(P7)按全局速度运动。
即：
MovL(P1)： 速度为 全局速度 的 直线运动
MovL(P2)： 速度为 焊接速度 的 直线运动
MovL(P3)： 速度为 焊接速度 的 摆弧运动
Arc(P4,P5)：速度为 焊接速度 的 摆弧运动
MoveL(P6)：速度为 焊接速度 的 直线运动
MoveL(P7)：速度为 全局速度 的 直线运动
备注：WeaveParams/WeaveStart/WeaveEnd指令只影响规划轨迹

3.107 RelPointWeldLine()

- 功能：多层多道功能使用的焊接轨迹直线偏移指令
- 格式：

  RelPointWeldLine(StartX，EndX，Y，Z，WorkAngle，TravelAngle，P1，P2)
- 支持端口：29999
- 参数列表

参数名	类型	含义	是否必填	默认值	参数范围
StartX	string	起点沿焊道坐标系X方向偏移距离	是		
EndX	string	终点沿焊道坐标系X方向偏移距离	是		
Y	string	起点和终点沿焊道坐标系Y方向的偏移距离	是		
Z	string	起点和终点沿焊道坐标系Z方向的偏移距离	是		
WorkAngle	string	焊枪沿焊道坐标系X方向旋转	是		
TravelAngle	string	焊枪沿焊道坐标系Y方向旋转	是		
joint = {j1, j2, j3, j4, j5, j6} OR pose= {x,y,z,rx,ry,rz}	string	直线运动指令起点	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 × 10^(308), 1.7976931348623157 × 10^(308))
joint = {j1, j2, j3, j4, j5, j6} OR	string	直线运动指令终点	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 ×

参数名	类型	含义	是否必填	默认值	参数范围
pose={x,y,z,rx,ry,rz}					10^(308), 1.7976931348623157 × 10^(308))

- 返回值：

  ErrorID,{start point:{x1,y1,z1,Rx1,Ry1,Rz1}end point:{x2,y2,z2,Rx2,Ry2,Rz2}},RelPointWeldLine()

  其中 x1,y1,z1,Rx1,Ry1,Rz1 是叠加计算偏移后的起点，以笛卡尔点位的形式提供。

  x2,y2,z2,Rx2,Ry2,Rz2 是叠加计算偏移后的终点，以笛卡尔点位的形式提供。

  示例:

  发送：

  relPointWeldLine(0,0,10,10,0,0,pose={455,-306,648,-176,-13,-101},pose={475,-306,648,-176,-13,-101})

  返回:

  0,{start point:{455.2566,-318.0591,640.6167,-176.0000,-13.0000,-101.0000}end point:{475.2566,-318.0591,640.6167,-176.0000,-13.0000,-101.0000}},relPointWeldLine(0,0,10,10,0,0,pose={455,-306,648,-176,-13,-101},pose={475,-306,648,-176,-13,-101});

3.108 RelPointWeldArc()

- 功能：多层多道功能使用的焊接轨迹圆弧偏移指令
- 格式：

  RelPointWeldArc(StartX，EndX，Y，Z，WorkAngle，TravelAngle，P1，P2，P3)

- 支持端口：29999
- 参数列表

参数名	类型	含义	是否必填	默认值	参数范围
StartX	string	起点沿焊道坐标系X方向偏移距离	是		
EndX	string	终点沿焊道坐标系X方向偏移距离	是		
Y	string	起点和终点沿焊道坐标系Y方向的偏移距离	是		
Z	string	起点和终点沿焊道坐标系Z方向的偏移距离	是		

参数名	类型	含义	是否必填	默认值	参数范围
WorkAngle	string	焊枪沿焊道坐标系X方向旋转	是		
TravelAngle	string	焊枪沿焊道坐标系Y方向旋转	是		
joint = {j1, j2, j3, j4, j5, j6} OR pose={x,y,z,rx,ry,rz}	string	圆弧指令起点	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 × 10^(308), 1.7976931348623157 × 10^(308))
joint = {j1, j2, j3, j4, j5, j6} OR pose={x,y,z,rx,ry,rz}	string	圆弧指令中间过渡点	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 × 10^(308), 1.7976931348623157 × 10^(308))
joint = {j1, j2, j3, j4, j5, j6} OR pose={x,y,z,rx,ry,rz}	string	圆弧指令结束点	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 × 10^(308), 1.7976931348623157 × 10^(308))

- 返回值：

  ErrorID,{start point:{x1,y1,z1,Rx1,Ry1,Rz1}middle point:{x2,y2,z2,Rx2,Ry2,Rz2}end point:{x3,y3,z3,Rx3,Ry3,Rz3}},RelPointWeldArc()

  其中 x1,y1,z1,Rx1,Ry1,Rz1 是叠加计算偏移后的起点，以笛卡尔点位的形式提供。

  x2,y2,z2,Rx2,Ry2,Rz2 是叠加计算偏移后的中间过渡点，以笛卡尔点位的形式提供。

  x3,y3,z3,Rx3,Ry3,Rz3 是叠加计算偏移后的中间过渡点，以笛卡尔点位的形式提供。

  示例:

  发送：

  relPointWeldArc(0,0,10,10,0,0,pose={455,-306,648,-176,-13,-101},pose={475,-326,648,-176,-13,-101},pose={{495,-306,648,-176,-13,-101})

  返回：

  0,{start point:{445.2600,-308.3359,638.0162,-176.0000,-13.0000,-101.0000}middle point:{475.2566,-338.0591,640.6167,-176.0000,-13.0000,-101.0000}end point:{505.2531,-308.3359,638.5439,-176.0000,-13.0000,-101.0000}},relPointWeldArc(0,0,10,10,0,0,pose={455,-306,648,-176,-13,-101},pose={475,-326,648,-176,-13,-101},pose={{495,-306,648,-176,-13,-101});

3.109 ArcTrackParams()

- 功能：设置跟踪过程中使用的一些参数
- 格式：

ArcTrackParams(sampleTime，coordinateType，upDownCompensationMin，upDownCompensationMax，upDownCompensationOffset，leftRightCompensationMin，leftRightCompensationMax，leftRightCompensationOffset)

- 支持端口：29999
- 参数列表

参数名	类型	含义	是否必填	参数范围
sampleTime	int	规划补偿运动的时长，有摆焊默认为1/2周期，无摆焊手动设置	是	0 ~ 10 000单位:mm
coordinateType	int	坐标系类型	是	1:user 用户坐标系2:tool 工具坐标系 3:weld 焊道坐标系
upDownCompensationMin	double	上下最小补偿量，沿Z向补偿限制，超过钳制	是	0 ~ 999单位:mm
upDownCompensationMax	double	上下最大补偿量，沿Z向补偿限制，超过钳制	是	0 ~ 9999单位:mm
upDownCompensationOffset	int	上下补偿偏移量	是	-100 ~ 100 %
leftRightCompensationMin	double	左右最小补偿量，沿Y向补偿限制，超过钳制	是	0 ~ 999单位:mm
leftRightCompensationMax	double	左右最大补偿量，沿Y向补偿限制，超过钳制	是	0 ~ 9999单位:mm
leftRightCompensationOffset	int	左右补偿偏移量	是	-100 ~ 100 %

- 返回值：

ErrorID,{},ArcTrackParams()

示例:

发送：

ArcTrackParams(250, 3, 0, 100, 0, 0, 100, 0)

返回:

0,{},ArcTrackParams(250, 3, 0, 100, 0, 0, 100, 0);

备注：请搭配ArcTrackStart()、SetArcTrackOffset()和ArcTrackEnd()使用，下方给出简单使用demo

3.110 ArcTrackStart**()**

- 功能：沿坐标系实时偏移指令开启
- 格式：

ArcTrackStart()

- 支持端口：29999
- 返回值：

ErrorID,{},ArcTrackStart()

示例:

发送：

ArcTrackStart()

返回:

0,{},ArcTrackStart();

备注：请搭配ArcTrackParams()、SetArcTrackOffset()和ArcTrackEnd()使用，下方给出简单使用demo

3.111 SetArcTrackOffset()

- 功能：设置沿坐标系实时偏移
- 格式：

SetArcTrackOffset({offsetX,offsetY,offsetZ,offsetRx,offsetRy,offsetRz})

- 支持端口：29999
- 参数列表

参数名	类型	含义	是否必填	参数范围
实时偏移量	double数组	补偿坐标系下实时偏移值	是	

- 返回值：

ErrorID,{},SetArcTrackOffset({offsetX,offsetY,offsetZ,offsetRx,offsetRy,offsetRz})

示例:

发送：

SetArcTrackOffset({0,5,0,0,0,0})

返回:

0,{},SetArcTrackOffset({0,5,0,0,0,0});

备注：请搭配ArcTrackParams()、ArcTrackStart()和ArcTrackEnd()使用，下方给出简单使用demo

3.112 ArcTrackEnd()

- 功能：沿坐标系实时偏移指令关闭
- 格式：

ArcTrackEnd()

- 支持端口：29999
- 返回值：

ErrorID,{},ArcTrackEnd()

示例:

发送：

ArcTrackEnd()

返回：

0,{},ArcTrackEnd();

备注：请搭配ArcTrackParams()、SetArcTrackOffset()和ArcTrackStart()使用，下方给出简单使用demo

跟踪相关的demo

```
MovL(P1)
WeldArcSpeed(10)
WeldArcSpeedStart()
MovL(P2)
WeaveParams(1,1,2,3,0,0,0,0,0,0)
WeaveStart()
ArcTrackParams(250,3,0,100,0,0,100,0)
ArcTrackStart()                                  子线程
---------------------------------------->       start
MovL(P3)                                         SetArcTrackOffset(0,5,0,0,0,0)
Arc(P4，P5)                                      SetArcTrackOffset(0,-5,0,0,0,0)
……                                                 ……
---------------------------------------->       finish
ArcTrackEnd()
WeaveEnd()
MoveL(P6)
WeldArcSpeedEnd()
MovL(P7)

含义：
在正常的摆焊过程中，先预设跟踪参数，指定叠加时间、坐标系和叠加偏移的钳制边界值等参数，然后通过队列指令开始实时跟踪。此时在子线程周期发送设置偏移的指令，实时叠加偏移。叠加偏移的值取相对值，如第一次下发(0,5,0,0,0,0),此时绝对偏移是(0,5,0,0,0,0),第二次下发(0,-5,0,0,0,0),此时绝对偏移是(0,0,0,0,0,0)。
```

3.113 SetResumeOffset()

- 功能：设置焊接过程中暂停，轨迹恢复的回退距离
- 格式：

SetResumeOffset(distance)

参数名	类型	含义	是否必填	默认值	参数范围
distance	double	设置暂停后，轨迹恢复时沿前进方向回退的距离，单位mm	是	0	

- 支持端口：29999
- 返回值：

ErrorID,{},SetResumeOffset()

示例:

发送：

SetResumeOffset(20)

返回：

0,{},setResumeOffset(20);

备注：

1、在weldArcSpeed有效时生效

2、需要先设置回退距离再暂停，才能正常规划回退点

3.114 pathRecovery()

- 功能：运动过程中暂停后，下发进行轨迹恢复，机器人回到暂停点
- 格式：

pathRecovery()

- 支持端口：29999
- 返回值：

ErrorID,{},SetResumeOffset()

示例:

发送：

pathRecovery()

返回：

0,{},pathRecovery();

备注：

1、该指令只会控制机器人回到暂停点，需要恢复运行需要另外下发Continue指令

2、该指令接口是异步接口，下发后立即返回。可以通过调用pathRecoveryStatus()来判断是否已经返回到了暂停点。

3.114 PathRecoveryStop()

- 功能：运动过程中暂停后，下发进行轨迹恢复，机器人回到暂停点的过程中，停止机器人。
- 格式：

pathRecoveryStop()

- 支持端口：29999
- 返回值：

ErrorID,{},pathRecoveryStop()

示例:

发送：

pathRecoveryStop()

返回：

0,{},pathRecoveryStop();

3.115 pathRecoveryStatus()

- 功能：运动过程中暂停后，下发进行轨迹恢复，机器人是否已经回到暂停点的状态返回
- 格式：

pathRecoveryStatus()

- 支持端口：29999
- 返回值：

ErrorID,{},pathRecoveryStatus()

示例:

发送：

pathRecoveryStatus()

返回:

0,{status},pathRecoveryStatus();

status:

0:已经回到暂停点;

1:不在暂停点，与暂停点有较小偏差;

2:不在暂停点，与暂停点有较大偏差；

3.116 EnableFTSensor()

- 功能：开启/关闭力传感器
- 格式：
参数名	类型	含义	是否必填	默认值	参数范围
status	int	0：关闭 1：开启	是		0/1

EnableFTSensor(status)

- 支持端口：29999
- 返回值：

ErrorID,{},EnableFTSensor(status)

示例:

发送：

EnableFTSensor(1)

返回:

0,{},EnableFTSensor(1);

3.117 SixForceHome()

- 功能：将力传感器当前数值置零
- 格式：
- 支持端口：29999
- 返回值：

ErrorID,{},SixForceHome()

示例:

发送：

SixForceHome()

返回:

0,{},SixForceHome();

3.118 GetForce()

- 功能：读取施加在TCP方向上的力和力矩
- 格式：
参数名	类型	含义	是否必填	默认值	参数范围
tool	int	X、Y、Z方向参考的坐标系	否	当前设置的工具坐标系值	[0,50]

GetForce(1)

- 支持端口：29999
- 返回值：

ErrorID,{x,y,z,rx,ry,rz},GetForce(1)

示例:

发送：

GetForce(1)

返回:

0,{x,y,z,rx,ry,rz},GetForce(1);

3.119 ForceDriveMode()

- 功能：设置并进入力控拖拽模式
- 格式：
参数名	类型	含义	是否必填	默认值	参数范围
{x,y,z,rx,ry,rz}	string	代表要释放的轴，0代表方向不能拖动，1代表方向可以拖动	是		0/1
user	int	设置参考的用户坐标系，基于用户坐标系进行拖拽	否	不填，则默认参考工具坐标系进行拖拽	[0,50]

ForceDriveMode({1,1,1,1,1,1})

- 支持端口：29999
- 返回值：

ErrorID,{},ForceDriveMode({1,1,1,1,1,1})

示例:

发送：

ForceDriveMode({1,1,1,1,1,1})

返回:

0,{},ForceDriveMode({1,1,1,1,1,1});

3.120 ForceDriveSpeed()

- 功能：力控拖拽速度百分比设置
- 格式：
参数名	类型	含义	是否必填	默认值	参数范围
speed	int	力控拖拽百分比	是		[1,100]

ForceDriveSpeed(10)

- 支持端口：29999
- 返回值：

ErrorID,{},ForceDriveSpeed(10)

示例:

发送：

ForceDriveSpeed(10)

返回:

0,{},ForceDriveSpeed(10);

3.121 FCForceMode

- 功能：以用户指定的配置参数开启力控
- 格式：

  FCForceMode({x,y,z,rx,ry,rz},{fx,fy,fz,frx,fry,frz},reference=0,user=0,tool=0)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必选	默认值	参数范围
{x,y,z,rx,ry,rz},	string	开启/关闭某个方向的力控调节	是		0/1
{fx,fy,fz,frx,fry,frz}	string	目标力	是		位移方向[-200,200]，单位N 姿态方向[-12,12]，单位N/m
reference=0	string	参考坐标系 0：参考工具坐标系 1：参考用户坐标系	否	0	0/1
user=0	string	选择已标定的用户坐标系	否	当前设置的用户坐标系值	[0,50]
tool=0	string	选择已标定的工具坐标系	否	当前设置的工具坐标系值	[0,50]

- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},FCForceMode({1,1,1,1,1,1},{100,100,100,10,10,10},reference=1,user=1);

- 示例：

  格式：FCForceMode({1,1,1,1,1,1},{100,100,100,10,10,10},reference=1,user=1)

3.122 FCSetDeviation

- 功能：设置力控模式下的位移和姿态偏差，若力控过程中恒力偏移了较大的距离，机器人进行相应处理
- 格式：

  FCSetDeviation({x,y,z,rx,ry,rz}，controltype)

- FCSetDeviation支持端口：29999
- 参数详解：

参数名	类型	含义	是否必选	默认值	参数范围
{x,y,z,rx,ry,rz}	string	分别代表力控模式下的位移和角度偏差阈值，前三个数单位为mm，后三个数单位为°	是	100/36	方向：(0,1000] 姿态：(0,360]
controltype	int	力控过程中超过了规定阈值时，机器人的处理方式 0：超过阈值则机器人报警 1：超过阈值则机器人停止搜寻而在原有轨迹上继续运动	否	0	0/1

- 返回：

ErrorID,{},FCSetDeviation({100,100,100,36,36,36});

- 示例：

  格式：FCSetDeviation({100,100,100,36,36,36})

- 注：FCSetDeviation 未调用时，设置各方向默认值为100，姿态36，TCP模式退出会恢复默认值

3.123 FCSetForceLimit

- 功能：设置最大力限制
- 格式：

  FCSetForceLimit(x,y,z,rx,ry,rz)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必选	默认值	参数范围
x	double	x方向的力限制	是	500	(0,500]
y	double	y方向的力限制	是	500	(0,500]
z	double	z方向的力限制	是	500	(0,500]
rx	double	rx方向的力限制	是	50	(0,50]
ry	double	ry方向的力限制	是	50	(0,50]
rz	double	rz方向的力限制	是	50	(0,50]

- 返回：

  ErrorID,{},FCSetForceLimit(500,500,500,20,20,20);

- 示例：

  格式：FCSetForceLimit(500,500,500,20,20,20)

- 注：FCSetForceLimit未调用时，设置各方向默认值为500，姿态50，TCP模式退出会恢复默认值

3.124 FCSetMass

- 功能：设置力惯性
- 格式：

  FCSetMass(x,y,z,rx,ry,rz)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必选	默认值	参数范围
x	double	x方向的力惯性	是	20	(0,10000]
y	double	y方向的力惯性	是	20	(0,10000]
z	double	z方向的力惯性	是	20	(0,10000]

参数名	类型	含义	是否必选	默认值	参数范围
rx	double	rx方向的力惯性	是	20	(0,10000]
ry	double	ry方向的力惯性	是	20	(0,10000]
rz	double	rz方向的力惯性	是	20	(0,10000]

- 返回：

  ErrorID,{},FCSetMass(20,20,20,20,20,20);

- 示例：

  格式：FCSetMass(20,20,20,20,20,20)

- 注：FCSetMass 未调用时，默认值是[20,20,20,20,20,20]，TCP模式退出会恢复默认值

3.125 FCSetStiffness

- 功能：设置力刚度
- 格式：

  FCSetStiffness(x,y,z,rx,ry,rz)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必选	默认值	参数范围
x	double	x方向的力刚度	是	30	[0,10000]
y	double	y方向的力刚度	是	30	[0,10000]
z	double	z方向的力刚度	是	30	[0,10000]
rx	double	rx方向的力刚度	是	30	[0,10000]
ry	double	ry方向的力刚度	是	30	[0,10000]
rz	double	rz方向的力刚度	是	30	[0,10000]

- 返回：

  ErrorID,{},FCSetStiffness(30,30,30,30,30,30);

- 示例：

  格式：FCSetStiffness(30,30,30,30,30,30)

- 注：FCSetStiffness未调用时，默认值是[30,30,30,30,30,30]，TCP模式退出会恢复默认值

3.126 FCSetDamping

- 功能：设置力阻尼
- 格式：

  FCSetDamping(x,y,z,rx,ry,rz)

- 支持端口：29999

- 参数详解：

参数名	类型	含义	是否必选	默认值	参数范围
x	double	x方向的力阻尼	是	50	[0,1000]
y	double	y方向的力阻尼	是	50	[0,1000]
z	double	z方向的力阻尼	是	50	[0,1000]
rx	double	rx方向的力阻尼	是	50	[0,1000]
ry	double	ry方向的力阻尼	是	50	[0,1000]
rz	double	rz方向的力阻尼	是	50	[0,1000]

- 返回：

  ErrorID,{},FCSetDamping(50,50,50,50,50,50);

- 示例：

  格式：FCSetDamping(50,50,50,50,50,50)

- 注：FCSetDamping未调用时，默认值是[50,50,50,50,50,50]，TCP模式退出会恢复默认值

3.127 FCOff

- 功能：关闭力控
- 格式：

  FCOff()

- 支持端口：29999
- 参数详解：
- 返回：ResultID:算法队列ID

  ErrorID,{ResultID},FCOff();

- 示例：

  格式：FCOff()

3.128 FCSetForceSpeedLimit

- 功能：力控调节速度限制
- 限制力控速度的目的，理论上受力足够大力控产生的速度可以无限大，有一定的风险性。设置较小的力控速度上限，力控调节速度较慢，适合低速平缓的接触面。设置较大的力控速度上限，力控调节速度快，适合高速力控应用。需要根据具体的应用场景进行调整。
- 格式：

  FCSetForceSpeedLimit(x,y,z,rx,ry,rz)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必选	默认值	参数范围
x,y,z,rx,ry,rz	double	x方向的力控调节速度限制	是	20	CRA机型： (0,安全限制TCP速度值] 其他机型： (0,300]
y		y方向的力控调节速度限制	是	20	CRA机型： (0,安全限制TCP速度值] 其他机型： (0,300]
z		z方向的力控调节速度限制	是	20	CRA机型： (0,安全限制TCP速度值] 其他机型： (0,300]
rx		rx方向的力控调节速度限制	是	20	CRA机型： (0,（4*安全限制TCP速度值 * 0.001 / 3.14 * 180）] 其他机型： (0,90]
ry		ry方向的力控调节速度限制	是	20	CRA机型： (0,（4*安全限制TCP速度值 * 0.001 / 3.14 * 180）] 其他机型： (0,90]
rz		rz方向的力控调节速度限制	是	20	CRA机型： (0,（4*安全限制TCP速度值 * 0.001 / 3.14 * 180）] 其他机型： (0,90]

ErrorID,{},FCSetForceSpeedLimit(20,20,20,20,20,20);

- 示例：

  格式：FCSetForceSpeedLimit(20,20,20,20,20,20)

- 注：FCSetForceSpeedLimit未调用时，默认值是[20,20,20,20,20,20]，TCP模式退出会恢复默认值

3.129 FCSetForce

- 功能：实时调整恒力设置
- 格式：

  FCSetForce(x,y,z,rx,ry,rz)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必选	默认值	参数范围
x	double	x方向的力值	是		[-200,200]，单位N
y	double	y方向的力值	是		[-200,200]，单位N
z	double	z方向的力值	是		[-200,200]，单位N
rx	double	rx方向的力值	是		[-12,12]，单位N/m
ry	double	ry方向的力值	是		[-12,12]，单位N/m
rz	double	rz方向的力值	是		[-12,12]，单位N/m

- 返回：

  ErrorID,{},FCSetForce(50,50,50,10,10,10);

- 示例：

  格式：FCSetForce(50,50,50,10,10,10)

3.130 RunTo

- 功能：点到点/直线运动，目标点位为关节/笛卡尔点位。
- 格式： **RunTo(joint = {j1, j2, j3, j4, j5, j6}, moveType = 1, user = 1, tool = 0, a = 20, v = 50)**
  RunTo(pose= {x,y,z,rx,ry,rz}, moveType = 0, user = 1, tool = 0, a = 20, v = 50)
- 支持端口：29999
- 参数详解：
-

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6} ORpose= {x,y,z,rx,ry,rz}	string	目标点的坐标值	是		joint值：任意值pose值：x/y/z任意值rx/ry/rz：(-1.7976931348623157 × 10^(308),1.7976931348623157 × 10^(308))
moveType=0	string	运动类型	否	直线运动	[0,4]int值 0：关节运动；1：直线运动；2：关节运动至指定偏移角度；3：沿工具坐标系进行相对直线运动；4：沿用户坐标系进行相对直线运动
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50]int值
tool=1	string	工具	否	上一次全	[0,50]int值

参数名	类型	含义	是否必填	默认值	参数范围
		坐标系索引		局指令设置值	
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值

- 返回：

  ErrorID,{},RunTo(pose= {x,y,z,rx,ry,rz},moveType = 1, user = 1, tool = 0, a = 20, v = 50)

- 示例：

  RunTo(joint = {1, 2, 3, 4, 5, 6},user = 1, moveType = 1, tool = 0, a = 20, v = 50)

- 返回：

  ErrorID,{},RunTo(joint = {1, 2, 3, 4, 5, 6},moveType = 1, user = 1, tool = 0, a = 20, v = 50);

3.131 PathRecovery

- 功能：从当前位置回到原轨迹暂停点，只有在暂停状态下才有效
- 格式： **PathRecovery()**
- 支持端口：29999
- 参数详解：无
- 返回：

  ErrorID,{},PathRecovery()

3.132 RequestControl

- 功能：请求将设备控制模式切换为TCP模式。
- 格式：

  RequestControl()

- 支持端口：29999

- 参数详解：
- 返回：

  ErrorID,{},RequestControl();

- 示例：

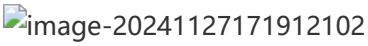
  格式：RequestControl()

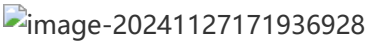
注：以下为能否切换TCP模式情况

控制器状态	是否允许切换TCP模式
未上电	允许
下使能（非暂停、非松抱闸）	允许
使能空闲	不允许
拖拽模式	不允许
单次运动中	不允许
运行中	不允许
暂停	不允许
错误（上使能情况下）	不允许
松抱闸	不允许
开启手自动模式	不允许

3.133 CreateTray

3.134 GetTrayPoint





3.135 FCCollisionSwitch

- 功能：力传感器碰撞检测功能开关
- 格式：

  FCCollisionSwitch(switch)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必选	默认值	参数范围
switch	int	传感器碰撞检测功能开关	是		[0,1]，为0时关闭力传感器碰撞检测功能，为1时开启力传感器碰撞检测功能

- 返回：

  ErrorID,{},FCCollisionSwitch(0);

- 示例：

  格式：FCCollisionSwitch(0)

3.136 SetFCCollision

- 功能：设置力控碰撞检测的阈值参数
- 格式：

  SetFCCollision(force,torque)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必选	默认值	参数范围
force	double	触发碰撞检测的力阈值	是		CR5:[5,150],CR10:[5,300],CR20[5,500]单位N
torque	double	触发碰撞检测的力矩阈值	是		CR5:[0.5,15],CR10:[0.5,30],CR20[0.5,50]，单位Nm

- 返回：

  ErrorID,{},FCSetForce(50,10);

- 示例：

  格式：FCSetForce(50,,10)

3.137 StartRTOffset

- 功能：启动坐标系偏移
- 格式：

  StartRTOffset()

- 支持端口：29999
- 参数详解：
- 返回：

  ErrorID,{},StartRTOffset();

- 示例：

格式：StartRTOffset()

3.138 EndRTOffset

- 功能：结束坐标系偏移
- 格式：

  EndRTOffset()
- 支持端口：29999
- 参数详解：
- 返回：

  ErrorID,{},EndRTOffset();
- 示例：

  格式：EndRTOffset()

3.139 OffsetPara

- 功能：设置坐标系偏移参数
- 格式：

  OffsetPara(x, y, z, rx, ry, rz)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
x	double	X轴方向偏移，单位：mm	是		任意值
y	double	Y轴方向偏移，单位：mm	是		任意值
z	double	Z轴方向偏移，单位：mm	是		任意值
rx	double	Rx 轴位置，单位：度	是		任意值
ry	double	Ry 轴位置，单位：度	是		任意值
rz	double	Rz 轴位置，单位：度	是		任意值

- 返回：

  ErrorID,{},OffsetPara(x, y, z, rx, ry, rz);
- 示例：

  格式：OffsetPara(10, 10, 10, 0, 0, 0)

3.140 CnvInit

- 功能：开启传送带，下发传送带配置信息，删除所有队列信息，并开始检测并存储新的队列信息
- 格式：

CnvInit(index)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
index	int	第几组参数	是		[1, 参数数量]

- 返回：

  ErrorID,{},CnvInit(index);

- 示例：

  格式：CnvInit(1)

3.141 GetCnvObject

- 功能：等待指定工件进入传送带进入抓取区域（拾取上边界与下边界组成的区域）
- 格式：

  GetCnvObject(objId)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
objId	int	工件类型, 0：不指定工件类型，获取最先入队的工件信息,对于传感器触发方式. 1~15：获取指定工件类型中最先入队的工件信息	是		[0, 15]

- 返回：
  - `flag`：类型integer
    - 0：没有工件
    - 1：有工件
    - -1：执行错误，重新执行
    - -2：有错误未处理
    - -3：非跟踪初始化状态，需执行CnvInit或StopSyncCnv
  - `objID`：类型integer，工件类型号
  - `objframe`：类型table，结果返回当前时刻的工作抓取坐标系，参考机器人基标系

  ErrorID,{flag, objId, objframe},GetCnvObject(objId);

- 示例：

  格式：GetCnvObject(0)

```
// TCP仿WaitCnvObject命令的伪代码说明:
while (true) {
    err, {flag, type, point} = TcpSendAndParse("GetCnvObject(0)")
        if (err) {
            // 错误，退出处理错误情况
            break;
```

```
        } else {
            // 成功
            if (flag == 0) {
                // 没有工件
                continue;
            } else if (flag == 1) {
                // 有工件，退出循环继续执行工艺其他命令
                break;
            } else if (flag == -1) {
                // 执行错误，继续调用
                continue;
            } else if (flag == -2) {
                // 当前机器人报错，需先处理错误
                break;
            } else if (flag == -3) {
                // 当前工艺未执行CnvInit，或者执行了StartSyncCnv需执行StopSyncCnv
                break;
            }
        }
}
```

3.142 StartSyncCnv

- 功能：开启同步带跟随
- 格式：

  StartSyncCnv()
- 支持端口：29999
- 参数详解：
- 返回：

  ErrorID,{},StartSyncCnv();
- 示例：

  格式：StartSyncCnv()

3.143 CnvMovL

- 功能：执行传动带跟随，采取直线轨迹插补
- 格式：

  CnvMovL(joint = {j1, j2, j3, j4, j5, j6},user = 1, tool = 0, a = 20, v = 50, cp = 100)

  CnvMovL(pose= {x,y,z,rx,ry,rz},user = 1, tool = 0, a = 20, v = 50, cp = 100)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6} OR pose= {x,y,z,rx,ry,rz}	string	点的坐标值	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 × 10^(308),

参数名	类型	含义	是否必填	默认值	参数范围
					1.7976931348623157 × 10^(308))
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值
r=20	string	过渡半径，单位mm	否		[0,100] int值

- 返回：
  - 跟随结果flag：类型integer
    - 0：执行成功
    - 1：跟随失败，未检测到工件类型
    - 2：跟随失败，已检测到工件类型，但未进入拾取边界范围
    - 3：跟随失败，工件超出离开边界

  ErrorID,{flag},CnvMovL(pose= {x,y,z,rx,ry,rz},user = 1, tool = 0, a = 20, v = 50, cp = 100) ;

- 示例：

  CnvMovL(pose= {x,y,z,rx,ry,rz},user = 1, tool = 0, a = 20, v = 50, cp = 100)

3.144 CnvMovC

- 功能：执行传动带跟随，采取圆弧轨迹插补
- 格式：

  CnvMovC(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6},user = 1, tool = 0, a = 20, v = 50, cp = 100, mode=1)

  CnvMovC(pose= {x,y,z,rx,ry,rz},pose= {x,y,z,rx,ry,rz},user = 1, tool = 0, a = 20, v = 50, cp = 100，mode=1)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
joint = {j1, j2, j3, j4, j5, j6} OR	string	表示圆弧中间点坐标值	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：(-1.7976931348623157 ×

参数名	类型	含义	是否必填	默认值	参数范围
pose={x,y,z,rx,ry,rz}					10^(308), 1.7976931348623157 × 10^(308))
joint = {j1, j2, j3, j4, j5, j6} OR pose={x,y,z,rx,ry,rz}	string	表示圆弧目标点坐标值	是		joint值：任意值 pose值：x/y/z任意值 rx/ry/rz：[-180,180]
user=1	string	用户坐标系索引	否	上一次全局指令设置值	[0,50] int值
tool=1	string	工具坐标系索引	否	上一次全局指令设置值	[0,50] int值
a=20	string	加速度百分比	否	上一次全局指令设置值	[1,100]int值
v=50	string	速度百分比	否	上一次全局指令设置值	[1,100]int值
cp=100	string	过渡百分比	否	上一次全局指令设置值	[0,100]int值
r=20	string	过渡半径，单位mm	否		>0 int值

- 返回：
  - 跟随结果flag：类型integer
    - 0：执行成功
    - 1：跟随失败，未检测到工件类型
    - 2：跟随失败，已检测到工件类型，但未进入拾取边界范围
    - 3：跟随失败，工件超出离开边界

  ErrorID,{flag},CnvMovC(joint = {j1, j2, j3, j4, j5, j6},joint = {j1, j2, j3, j4, j5, j6},user = 1, tool = 0, a = 20, v = 50, cp = 100);

- 示例：

  CnvMovC(joint = {1, 2, 3, 4, 5, 6},joint = {7, 8, 9, 10, 11, 12},user = 1, tool = 0, a = 20, v = 50, cp = 100)

3.145 StopSyncCnv

- 功能：停止同步传送带，运行完该条指令后才会继续下发执行该指令后的其他指令
- 格式：

  StopSyncCnv()

- 支持端口：29999
- 参数详解：
- 返回：

  ErrorID,{},StopSyncCnv();

- 示例：

  格式：StopSyncCnv()

3.146 SetCnvPointOffset

- 功能：设置传送带用户坐标系下X、Y方向的偏移
- 格式：

  SetCnvPointOffset(xOffset, yOffset)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
xOffset	double	X轴方向偏移	是		任意值
yOffset	double	Y轴方向偏移	是		任意值

- 返回：

  ErrorID,{},SetCnvPointOffset(xOffset, yOffset);

- 示例：

  格式：SetCnvPointOffset(10, 10)

3.147 SetCnvTimeCompensation

- 功能：设置时间补偿，用于补偿拍照触发带来的时间延时导致传送带运动方向的抓取位置偏移
- 格式：

  SetCnvTimeCompensation(time)

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
time	int	补偿时间（ms）	是		任意值

- 返回：

  ErrorID,{},SetCnvTimeCompensation(time);

- 示例：

  格式：SetCnvTimeCompensation(100)

3.148 MovS

- 功能：轨迹拟合功能
- 格式：

**MovS(point1,point2,point3,... ,tool =0,user=0,v=100,speed=100,a=100,freq=1) ** 或

**MovS(file=xxxxx.csv,tool =0,user=0,v=100,speed=100,a=100,freq=1) **

- 支持端口：29999
- 参数详解：

参数名	类型	含义	是否必填	默认值	参数范围
pointx	点位	待拟合的点位，可以是关节点位或者是位姿点位，点位数量范围是4~50个	否	无	
tool	int	进行拟合的点位所在的工具坐标系，可选参数的优先级为最高的优先级	否	无(文件内的点位可能包含坐标系)	[0,50]
user	int	进行拟合的点位所在的用户坐标系，可选参数的优先级为最高的优先级	否	无(文件内的点位可能包含坐标系)	[0,50]
v	double	速率百分比	否	上一次全局指令设置值	[1,100]
speed	double	绝对速度，同其他的运动指令	否		
a	double	加速度百分比	否	上一次全局指令设置值	[1,100]
freq	double	默认1，取值范围是（0,1] 需要客户根据输入的点位的平滑程度来适当的设置值。（超出范围控制器脚本接口要报错）数值越小，拟合之后曲线越光滑，但是实际点位和给定点位之间的偏差值会越大。 距离： CAD输出的轨迹可以保持为1，保证精度，如果是3D相机等的曲线，建议打开，保证曲线的平滑。	否	1	(0,1]
file	string	文件名字	否	无	

- 返回：

  **ErrorID,{}, MovS(pose={100,0,100,0,0,0},pose={100,20,100,0,0,0},pose={100,30,100,0,0,0}, pose={100,40,100,0,0,0}) **

- 示例：

  **格式： MovS(pose={100,0,100,0,0,0},pose={100,20,100,0,0,0},pose={100,30,100,0,0,0}，pose={100,40,100,0,0,0}) **

3.149 GetDOGroupDEC

- 功能：获取一组数字输出组端口状态，将组DO电平按0/1组成一个二进制数，再转化为十进制输出
- 格式：

  GetDOGroupDEC({index1, ..., indexn})
- 支持端口：29999
- 参数详解：
参数名	类型	含义	是否必填	参数范围
index	int	输出端口索引	是	[1,24]
- 返回：

  ErrorID,{value},GetDOGroup({index1,index2,index3,...});
- 示例：

GetDOGroup({1,2,3})

读取DO1、DO2和DO3的电平值，若DO1为高电平、DO2为低电平、DO3为低电平，则组成二进制数001，转化为十进制数为1。若DO1为低电平、DO2为低电平、DO3为高电平，则组成二进制数100，转化为十进制数为4。

3.150 DOGroupDEC（队列指令）

- 功能：设置输出组端口状态，给定一个十进制值，将该值转换为二进制值，按bit对应给定输入索引的DO值。
- 格式：

  DOGroup({index1,index2,...,indexn},value)
- 参数数量：不固定(最大支持24个)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	参数范围
index1	int	设置数字输出索引1	[1,24]
index2	int	设置数字输出索引2	[1,24]
...	...	...	...
indexn	int	设置数字输出索引n	[1,24]
value	int	设置数字输出端口状态	>0 int值
- 返回： ResultID:算法队列ID

  ErrorID,{ResultID},DOGroup({index1,index2,...,indexn},value);
- 示例：

  DOGroup({1,2,3,4}, 11)

  将11转化为2进制数，即0b1011，则bit0的1为给到DO1的输出，bit1的1为给到DO2的输出，bit2的0为给到DO3的输出，bit3的1为给到DO4的输出，最终结果为DO1输出高电平，DO2输出高电平，DO3输出低电平，DO4输出高电平。

3.151 DIGroupDEC

- 功能：获取输入组端口状态，将组DI电平按0/1组成一个二进制数，再转化为十进制输出
- 格式：

  DIGroup({index1,index2,...,indexn})
- 参数数量：不固定(最大支持24个)
- 支持端口：29999
- 参数详解：

参数名	类型	含义	参数范围
index1	int	数字输入索引	[1,24]
...	...	...	...
indexn	int	数字输入索引	[1,24]

- 返回：

  ErrorID,{value},DIGroup({index1,index2,...,indexn});
- 示例：

  GetDOGroup({1,2,3})

  读取DI1、DI2和DI3的电平值，若DI1为高电平、DI2为低电平、DI3为低电平，则组成二进制数001，转化为十进制数为1。若DI1为低电平、DI2为低电平、DI3为高电平，则组成二进制数100，转化为十进制数为4。

4.实时反馈端口

30004服务器端口，**每8ms**能收到一次机器人反馈信息

30005服务器端口**每200ms**能收到一次机器人反馈信息

30006端口为**可配置**的反馈机器人信息端口(默认为每**50ms**反馈)，30006端口的实时数据的配置更新可以在线修改后，实时生效；

通过反馈端口每次收到的数据包有1440个字节，这些字节以标准的格式排列，数据以小端方式储存。

意义/Meaning	值的数目/Number of values	数据类型/Type	描述/Notes	字节大小/Size in bytes	字节位置值/Byte position valueCR	支持产品	是否实现
MessageSize	1	unsigned short	消息字节总长度/Total message length in bytes	2	0000 ~ 0001	CR/Nova/ED6	√
reserved	3	unsigned short	保留位	6	0002 ~ 0007	CR/Nova/ED6	
DigitalInputs	1	uint64	数字输入/Current state	8	0008 ~ 0015	CR/Nova/ED6	√

意义/Meaning	值的数目/Number of values	数据类型/Type	描述/Notes	字节大小/Size in bytes	字节位置值/Byte position valueCR	支持产品	是否实现
			of the digital inputs.				
DigitalOutputs	1	uint64	数字输出	8	0016 ~ 0023	CR/Nova/ED6	√
RobotMode	1	uint64	机器人模式/Robot mode	8	0024 ~ 0031	CR/Nova/ED6	√
TimeStamp	1	uint64	机器人系统时间戳	8	0032 ~ 0039	CR/Nova/ED6	√
RunTime	1	uint64	机器人开机运行时间（单位ms）	8	0040 ~ 0047	CR/Nova/ED6	√
TestValue	1	uint64	内存结构测试标准值 0x0123 4567 89AB CDEF	8	0048 ~ 0055	CR/Nova/ED6	√
****	1	double	保留位	8	0056 ~ 0063	CR/Nova/ED6	
SpeedScaling	1	double	速度比例/Speed scaling of the trajectory limiter	8	0064 ~ 0071	CR/Nova/ED6	√
LinearMomentumNorm	1	double	机器人当前动量/Norm of Cartesian linear momentum(需要特定硬件版本)	8	0072 ~ 0079	CR/Nova/ED6	×
VMain	1	double	控制板电压/Masterboard: Main voltage	8	0080 ~ 0087	CR/Nova/ED6	×
VRobot	1	double	机器人电压/Masterboard: Robot voltage (48V)	8	0088 ~ 0095	CR/Nova/ED6	√
IRobot	1	double	机器人电流/Masterboard: Robot current	8	0096 ~ 0103	CR/Nova/ED6	√
ProgramState	1	double	脚本运行状态	8	0104 ~ 0111	CR/Nova/ED6	√
SafetyIOIn	2	char	安全IO输入状态	2	0112 ~ 0113	CR/Nova/ED6	√
SafetyIOOut	2	char	安全IO输出状态	2	0114 ~ 0115	CR/Nova/ED6	√
	4	char	保留	4	0116 ~ 0119	CR/Nova/ED6	×

意义/Meaning	值的数目/Number of values	数据类型/Type	描述/Notes	字节大小/Size in bytes	字节位置值/Byte position valueCR	支持产品	是否实现
ToolAcceleroMeter	3	double	TCP加速度/Tool x,y and z accelerometer values(需要特定硬件版本)	24	0120 ~ 0143	CR/Nova/ED6	×
ElbowPosition	3	double	肘位置/Elbow position(需要特定硬件版本)	24	0144 ~ 0167	CR/Nova/ED6	×
ElbowVelocity	3	double	肘速度/Elbow velocity(需要特定硬件版本)	24	0168 ~ 0191	CR/Nova/ED6	×
QTarget	6	double	目标关节位置/Target joint positions	48	0192 ~ 0239	CR/Nova/ED6	√
QDTarget	6	double	目标关节速度/Target joint velocities	48	0240 ~ 0287	CR/Nova/ED6	√
QDDTarget	6	double	目标关节加速度/Target joint accelerations	48	0288 ~ 0335	CR/Nova/ED6	√
ITarget	6	double	目标关节电流/Target joint currents	48	0336 ~ 0383	CR/Nova/ED6	√
MTarget	6	double	目标关节扭矩/Target joint moments (torques)	48	0384 ~ 0431	CR/Nova/ED6	√
QActual	6	double	实际关节位置/Actual joint positions	48	0432 ~ 0479	CR/Nova/ED6	√
QDActual	6	double	实际关节速度/Actual joint velocities	48	0480 ~ 0527	CR/Nova/ED6	√
IActual	6	double	实际关节电流/Actual joint currents	48	0528 ~ 0575	CR/Nova/ED6	√
ActualTCPForce	6	double	TCP传感器力值(通过六维力计算)	48	0576 ~ 0623	CR/Nova/ED6	√
ToolVectorActual	6	double	TCP笛卡尔实际坐标值/Actual Cartesian coordinates of the tool: (x,y,z,rx,ry,rz), where rx, ry and	48	0624 ~ 0671	CR/Nova/ED6	√

意义/Meaning	值的数目/Number of values	数据类型/Type	描述/Notes	字节大小/Size in bytes	字节位置值/Byte position valueCR	支持产品	是否实现
			rz is a rotation vector representation of the tool orientation				
TCPSpeedActual	6	double	TCP笛卡尔实际速度值/Actual speed of the tool given in Cartesian coordinates	48	0672 ~ 0719	CR/Nova/ED6	√
TCPForce	6	double	TCP力值（通过关节电流计算）	48	0720 ~ 0767	CR/Nova/ED6	√
ToolVectorTarget	6	double	TCP笛卡尔目标坐标值/Target Cartesian coordinates of the tool: (x,y,z,rx,ry,rz), where rx, ry and rz is a rotation vector representation of the tool orientation	48	0768 ~ 0815	CR/Nova/ED6	√
TCPSpeedTarget	6	double	TCP笛卡尔目标速度值/Target speed of the tool given in Cartesian coordinates	48	0816 ~ 0863	CR/Nova/ED6	√
MotorTemperatures	6	double	关节温度/Temperature of each joint in degrees celsius	48	0864 ~ 0911	CR/Nova/ED6	√
JointModes	6	double	关节控制模式/Joint control modes	48	0912 ~ 0959	CR/Nova/ED6	√
VActual	6	double	关节电压/Actual joint voltages	48	960 ~ 1007	CR/Nova/ED6	√
HandType	4	char	手系(备用参数)	4	1008 ~ 1011	CR/Nova/ED6	√
User	1	char	全局用户坐标系索引	1	1012	CR/Nova/ED6	√

意义/Meaning	值的数目/Number of values	数据类型/Type	描述/Notes	字节大小/Size in bytes	字节位置值/Byte position valueCR	支持产品	是否实现
Tool	1	char	全局工具坐标系索引	1	1013	CR/Nova/ED6	√
RunQueuedCmd	1	char	算法队列运行标志	1	1014	CR/Nova/ED6	√
PauseCmdFlag	1	char	算法队列暂停标志	1	1015	CR/Nova/ED6	√
VelocityRatio	1	char	关节速度比例(0~100)	1	1016	CR/Nova/ED6	√
AccelerationRatio	1	char	关节加速度比例(0~100)	1	1017	CR/Nova/ED6	√
JerkRatio	1	char	关节加加速度比例(0~100)	1	1018	CR/Nova/ED6	×
XYZVelocityRatio	1	char	笛卡尔位置速度比例(0~100)	1	1019	CR/Nova/ED6	√
RVelocityRatio	1	char	笛卡尔姿态速度比例(0~100)	1	1020	CR/Nova/ED6	√
XYZAccelerationRatio	1	char	笛卡尔位置加速度比例(0~100)	1	1021	CR/Nova/ED6	√
RAccelerationRatio	1	char	笛卡尔姿态加速度比例(0~100)	1	1022	CR/Nova/ED6	√
XYZJerkRatio	1	char	笛卡尔位置加加速度比例(0~100)	1	1023	CR/Nova/ED6	×
RJerkRatio	1	char	笛卡尔姿态加加速度比例(0~100)	1	1024	CR/Nova/ED6	×
BrakeStatus	1	char	机器人抱闸状态	1	1025	CR/Nova/ED6	√
EnableStatus	1	char	机器人使能状态	1	1026	CR/Nova/ED6	√
DragStatus	1	char	机器人拖拽状态 0：不在拖拽状态 1：关节拖拽状态 2：力控拖拽状态	1	1027	CR/Nova/ED6	√
RunningStatus	1	char	机器人运行状态(机器人是否在运动)	1	1028	CR/Nova/ED6	√
ErrorStatus	1	char	机器人报警状态	1	1029	CR/Nova/ED6	√
JogStatusCR	1	char	机器人点动状态	1	1030	CR/Nova/ED6	√
CRRobotType	1	char	机器类型	1	1031	CR/Nova/ED6	√
DragButtonSignal	1	char	按钮板拖拽信号	1	1032	CR/Nova/ED6	√
EnableButtonSignal	1	char	按钮板使能信号	1	1033	CR/Nova/ED6	√
RecordButtonSignal	1	char	按钮板录制信号	1	1034	CR/Nova/ED6	√
ReappearButtonSignal	1	char	按钮板复现信号	1	1035	CR/Nova/ED6	√

意义/Meaning	值的数目/Number of values	数据类型/Type	描述/Notes	字节大小/Size in bytes	字节位置值/Byte position valueCR	支持产品	是否实现
JawButtonSignal	1	char	按钮板夹爪控制信号	1	1036	CR/Nova/ED6	√
SixForceOnline	1	char	六维力在线状态: 0：离线 1：在线 2：异常	1	1037	CR/Nova/ED6	×
CollisionState	1	char	碰撞状态	1	1038	CR/Nova/ED6	√
ArmApproachState	1	char	小臂接近暂停状态	1	1039	CR/Nova/ED6	√
J4ApproachState	1	char	J4接近暂停	1	1040	CR/Nova/ED6	√
J5ApproachState	1	char	J5接近暂停	1	1041	CR/Nova/ED6	√
J6ApproachState	1	char	J6接近暂停	1	1042	CR/Nova/ED6	√
Reserve2[61]	1	char	保留位	61	1043-1103	CR/Nova/ED6	√
VibrationDisZ	1	double	加速度计测量Z轴抖动位移	8	1104~1111	CR/Nova/ED6	√
CurrentCommandId	1	uint64	当前运动队列id	8	1112~1119	CR/Nova/ED6	√
MActual[6]	6	double	实际扭矩	48	1120 ~ 1167	CR/Nova/ED6	√
Load	1	double	负载重量kg	8	1168-1175	CR/Nova/ED6	√
CenterX	1	double	X方向偏心距离mm	8	1176-1183	CR/Nova/ED6	√
CenterY	1	double	Y方向偏心距离mm	8	1184-1191	CR/Nova/ED6	√
CenterZ	1	double	Z方向偏心距离mm	8	1192-1199	CR/Nova/ED6	√
User[6]	6	double	用户坐标值	48	1200-1247	CR/Nova/ED6	√
Tool[6]	6	double	工具坐标值	48	1248-1295	CR/Nova/ED6	√
TraceIndex	1	double	轨迹复现运行索引	8	1296-1303	CR/Nova/ED6	×
SixForceValue[6]	6	double	当前六维力数据原始值	48	1304-1351	CR/Nova/ED6	√
TargetQuaternion[4]	4	double	[qw,qx,qy,qz] 目标四元数	32	1352-1383	CR/Nova/ED6	√
ActualQuaternion[4]	4	double	[qw,qx,qy,qz] 实际四元数	32	1384-1415	CR/Nova/ED6	√
AutoManualMode	1	char	手自动模式	2	1416~1417	CR/Nova/ED6	√
ExportStatus	1	unsigned short	U盘导出状态	2	1418~1419	CR/Nova/ED6	√
SafetyState	1	char	安全状态	1	1420	CR/Nova/ED6	√

意义/Meaning	值的数目/Number of values	数据类型/Type	描述/Notes	字节大小/Size in bytes	字节位置值/Byte position valueCR	支持产品	是否实现
			急停状态（低有效）	1bit	1420:0	CR/Nova/ED6	√
			防护性停止状态（低有效）	1bit	1420:1	CR/Nova/ED6	√
			缩减模式状态（低有效）	1bit	1420:2	CR/Nova/ED6	√
			非停止状态（低有效）	1bit	1420:3	CR/Nova/ED6	√
			运动中状态（低有效）	1bit	1420:4	CR/Nova/ED6	√
			系统急停状态（低有效）	1bit	1420:5	CR/Nova/ED6	√
			用户急停状态（低有效）	1bit	1420:6	CR/Nova/ED6	√
			安全原点输出状态（低有效(不在安全原点时有效)）	1bit	1420:7	CR/Nova/ED6	√
SafeState Reserve	1	char	安全状态保留位	1	1421	CR/Nova/ED6	
Reserve3[18]	1	char	保留位	18	1422 ~ 1439	CR/Nova/ED6	
TOTAL			1440byte package	1440			

说明：

- RobotMode返回机器人模式：
- BrakeStatus抱闸状态：

  0x01表示第六个轴抱闸打开；

  0x02表示第五个轴抱闸打开；

  0x03表示五六轴抱闸打开；

  0x04表示第四个轴抱闸打开；

  ...

7	6	5	4	3	2	1	0
保留位	保留位	关节一	关节二	关节三	关节四	关节五	关节六

- JointModes关节控制模式：

  当前值为8表示位置模式；

  当前值为10表示力矩模式；

- RobotType表示机器类型：

RobotType值	代表机型
3	CR3
113	CR3A
5	CR5
115	CR5A
7	CR7
117	CR7A
10	CR10
120	CR10A
12	CR12
122	CR12A
16	CR16
126	CR16A
130	CR20A
101	Nova2
103	Nova5
150	Magician E6

5.错误码描述

错误码	描述	备注
0	无错误	下发成功
-1	没有执行成功	命令执行失败
-2	当前处于错误状态拒绝执行指令	需要清除错误
-3	当前处于急停拍下状态拒绝执行指令	需要松开急停并清除错误
-4	当前处于下电状态拒绝执行指令	需要上电
-5	当前处于脚本运行状态拒绝执行指令	需要暂停/停止脚本
-6	MoveJog指令运动轴与运动类型不匹配	需调整指令coordtype参数值
-7	当前处于脚本暂停状态拒绝执行指令	需要停止脚本
-8	当前机器人的认证已经过期	机器人当前TCP处于不可用状态，需要联系FAE解决。
。。。	。。。	。。。
-10000	命令错误	不存在下发的命令
-20000	参数数量错误	下发命令中的参数数量错误
-30001	任意 带名称的必选参数 的参数类型错误	-30001表示带名称的必选参数类型错误：例如：joint="ss"

错误码	描述	备注
-30001	不带名称的必选参数 第一个参数的参数类型错误	-3000-表示必选参数类型错误 最后一位1表示下发第1个参数的参数类型错误
-30002	不带名称的必选参数 第二个参数的参数类型错误	-3000-表示必选参数类型错误 最后一位2表示下发第2个参数的参数类型错误
。。。	。。。	。。。
-40001	任意 带名称的必选参数 的参数范围错误	-40001表示带名称的必选参数范围错误：例如：pose={10,10,10,190,190,190}
-40001	不带名称的必选参数 第一个参数的参数范围错误	-4000-表示必选参数范围错误 最后一位1表示下发第1个参数的参数范围错误
-40002	不带名称的必选参数 第二个参数的参数范围错误	-4000-表示必选参数范围错误 最后一位2表示下发第2个参数的参数范围错误
。。。	。。。	。。。
-50001	任意 带名称可选参数 的参数类型错误	-50001表示带名称的可选参数类型错误：例如：use=ttt
-50001	不带名称的可选参数 第一个参数的参数类型错误	-5000-表示可选参数 参数类型错误 最后一位1表示下发第1个参数的参数类型错误
-50002	不带名称的可选参数 第二个参数的参数类型错误	-5000-表示可选参数 参数类型错误 最后一位2表示下发第2个参数的参数类型错误
。。。	。。。	。。。
-60001	任意带名称可选参数的 参数范围错误	-60001表示带名称可选参数的范围错误：例如：user=1000
-60001	不带名称的可选参数 第一个参数的参数范围错误	-6000-表示可选参数参数范围错误 最后一位1表示下发第1个参数的参数范围错误
-60002	不带名称的可选参数 第二个参数的参数范围错误	-6000-表示可选参数参数范围错误 最后一位2表示下发第2个参数的参数范围错误
。。。	。。。	。。。
-100001	名字太长	TCP创建托盘指令的错误码
-100002	点位的数量错误	TCP创建托盘指令的错误码
-100003	托盘配置错误	TCP创建托盘指令的错误码
-100004	托盘数量太多	TCP创建托盘指令的错误码

注：错误状态可执行指令：错误清除、错误查询、急停、机器人状态；

急停状态可执行指令：错误清除、错误查询、急停松开、机器人状态

下电状态可执行指令：错误清除、错误查询、上电、机器人状态、急停松开

注2： (2024-02-02 补充)

目前已知当TCP指令参数的内的 } 到下一个 , 之间的数据会因为定位不清晰被错误的分解，进而会导致参数数量和预期不符。 举例说明： TCP模式 下发指令 SetUser(1,{0,0,100,0,0,0}asdasdasd,1)， 会报错 -20000（参数数量错误） 而不是 -30002 (必选参数2 类型错误)。

注3：(2025-06-03补充)

TCP指令参数按 ‘,’ 解析，如果逗号前没有参数字符串，则按空字符捕获

举例说明：

TCP模式 下发指令 DIGroupDEC({1, ,8})，捕获3个参数，第一个参数为1，第二个参数为空字符，第三个参数为8，由于第二个参数空字符不是整形数，因此报错-30001.

TCP模式 下发指令 CalcTool(,0,{10,20,30,0,0,0})，捕获3个参数，第一个参数为空字符，第二个参数为0，第三个参数为{10,20,30,0,0,0}，由于第一个参数空字符不是整形数，因此报错-30001.

6. 各状态下允许执行的TCP指令

急停状态	错误状态	下电状态	脚本运行状态	脚本暂停状态	其余状态
ClearError()	ClearError()	ClearError()	SpeedFactor()	同脚本运行	无限制，指令均可发送
GetErrorID()	GetErrorID()	GetErrorID()	RobotMode()	同脚本运行	。。。
EmergencyStop()	EmergencyStop()	PowerOn()	DoInstant()	同脚本运行	
Stop()	Stop()	RobotMode()	ToolDoInstant()	同脚本运行	
RobotMode()	RobotMode()	Stop()	AOInstant()	同脚本运行	
LogExportUSB()	LogExportUSB()	EmergencyStop()	Stop()	同脚本运行	
GetExportStatus()	GetExportStatus()	LogExportUSB()	Pause()	同脚本运行	
		GetExportStatus()	Continue()	同脚本运行	
			GetStartPose()	同脚本运行	
			PositiveKin()	同脚本运行	
			InverseSolution()	同脚本运行	
			GetAngle()	同脚本运行	
			GetPose()	同脚本运行	
			EmergencyStop()	同脚本运行	
			ModbusCreate()	同脚本运行	
			ModbusClose()	同脚本运行	
			GetInBits()	同脚本运行	
			GetInRegs()	同脚本运行	
			GetCoils()	同脚本运行	
			SetCoils()	同脚本运行	
			GetHoldRegs()	同脚本运行	
			SetHoldRegs()	同脚本运行	
			GetErrorID()	同脚本运行	
			DI()	同脚本运行	
			ToolDI()	同脚本运行	
			AI()	同脚本运行	
			ToolAI()	同脚本运行	
			DIGroup()	同脚本运行	
			GetDO()	同脚本运行	

急停状态	错误状态	下电状态	脚本运行状态	脚本暂停状态	其余状态
			GetAO()	同脚本运行	
			GetDOGroup()	同脚本运行	
			SetTool485()	同脚本运行	
			SetToolPower()	同脚本运行	
			SetToolMode()	同脚本运行	
			CalcUser()	同脚本运行	
			CalcTool()	同脚本运行	
			GetInputBool()	同脚本运行	
			GetInputInt()	同脚本运行	
			GetInputFloat()	同脚本运行	
			GetOutputBool()	同脚本运行	
			GetOutputInt()	同脚本运行	
			GetOutputFloat()	同脚本运行	
			SetOutputBool()	同脚本运行	
			SetOutputInt()	同脚本运行	
			SetOutputFloat()	同脚本运行	
			GetCurrentCommandId()	同脚本运行	
			LogExportUSB()	同脚本运行	
			GetExportStatus()	同脚本运行	
			ClearError()	同脚本运行	
			GetForce()	同脚本运行	
				EnableRobot()	
				DisableRobot()	
				RunTo()	
				PathRecovery()	
				StartDrag()	
				StopDrag()	
				SixForceHome()	
				EnableFTSensor()	
				ForceDriveMode()	
				ForceDriveSpeed()	

注：针对RequestControl()指令的使用限定，详见指令说明