7. 机器人轨迹复现

7.1. 设置轨迹记录参数

`SetTPDParam(name, period_ms, type=1,di_choose=0, do_choose=0)`

设置轨迹记录参数

- `name`：轨迹名；
- `period_ms`：采样周期，固定值，2ms 或 4ms 或 8ms;

- `type`：数据类型，1–关节位置；
- `di_choose`：DI 选择,bit0~bit7 对应控制箱 DI0~DI7，bit8~bit9 对应末端DI0~DI1，0–不选择，1–选
- `do_choose`：DO 选择,bit0~bit7 对应控制箱 DO0~DO7，bit8~bit9 对应末端 DO0~DO1，0–不选择，

错误码 成功–0 失败– errcode

7.1.1. 代码示例

```python
from fairino import Robot
import time
# 与机器人控制器建立连接，连接成功返回一个机器人对象
robot = Robot.RPC('192.168.58.2')
type = 1  # 数据类型，1–关节位置
name = 'tpd2023'  # 轨迹名
period = 4  #采样周期，2ms或4ms或8ms
di = 0 # di输入配置
do = 0 # do输出配置
ret = robot.SetTPDParam(name, period, di_choose=di)    #配置TPD参数
print("配置TPD参数错误码", ret)
robot.Mode(1)  # 机器人切入手动模式
time.sleep(1)
robot.DragTeachSwitch(1)  #机器人切入拖动示教模式
ret = robot.GetActualTCPPose()
print("获取当前工具位姿", ret)
time.sleep(1)
ret = robot.SetTPDStart(name, period, do_choose=do)   # 开始记录示教轨迹
print("开始记录示教轨迹错误码", ret)
time.sleep(15)
ret = robot.SetWebTPDStop()  # 停止记录示教轨迹
print("停止记录示教轨迹错误码", ret)
robot.DragTeachSwitch(0)  #机器人切入非拖动示教模式
# robot.SetTPDDelete('tpd2023')   # 删除TPD轨迹
```

7.2. 开始轨迹记录

`SetTPDStart(name, period_ms, type=1,di_choose=0, do_choose=0)`

干始轨迹记录

- `name`：轨迹名；
- `period_ms`：采样周期，固定值，2ms或4ms或8ms；

- `type`：数数据类型，1–关节位置 默认1;
- `di_choose`：DI 选择,bit0~bit7 对应控制箱 DI0~DI7，bit8~bit9 对应末端DI0~DI1，0–不选择，1–选
- `do_choose`：DO 选择,bit0~bit7 对应控制箱 DO0~DO7，bit8~bit9 对应末端 DO0~DO1，0–不选择，

昔误码 成功–0 失败– errcode

7.3. 停止轨迹记录

原型	`SetWebTPDStop()`
描述	停止轨迹记录
必选参数	无
默认参数	无
返回值	错误码 成功–0 失败– errcode

7.4. 删除轨迹记录

原型	`SetTPDDelete(name)`
描述	删除轨迹记录
必选参数	• `name`:轨迹名
默认参数	无
返回值	错误码 成功–0 失败– errcode

7.5. 轨迹预加载

原型	`LoadTPD(name)`
描述	轨迹预加载
必选参数	• `name`:轨迹名
默认参数	无
返回值	错误码 成功–0 失败– errcode

7.5.1. 代码示例

```
from fairino import Robot
import time
# 与机器人控制器建立连接，连接成功返回一个机器人对象
robot = Robot.RPC('192.168.58.2')
# P1=[-321.821, 125.694, 282.556, 174.106, -15.599, 152.669]
name = 'tpd2023'    #轨迹名
blend = 1    #是否平滑，1-平滑，0-不平滑
ovl = 100.0    #速度缩放
ret = robot.LoadTPD(name)  #轨迹预加载
print("轨迹预加载错误码",ret)
ret,P1 = robot.GetTPDStartPose(name)    #获取轨迹起始位姿
print ("获取轨迹起始位姿错误码",ret,"起始位姿",P1)
ret = robot.MoveL(P1,0,0)        #运动到起始点
print("运动到起始点错误码",ret)
time.sleep(10)
ret = robot.MoveTPD(name, blend, ovl)  #轨迹复现
print("轨迹复现错误码",ret)
```

7.6. 获取轨迹起始位姿

原型	`GetTPDStartPose(name)`
描述	获取轨迹起始位姿
必选参数	• `name`:轨迹名
默认参数	无
返回值	• 错误码 成功-0 失败- errcode • `desc_pose=[x,y,z,rx,ry,rz]`：轨迹起始位姿

7.7. 轨迹复现

原型	`MoveTPD(name,blend,ovl)`
描述	轨迹复现
必选参数	• `name`:轨迹名 • `blend`：是否平滑，0-不平滑，1-平滑 • `ovl`：速度缩放因子，范围[0~100]
默认参数	无
返回值	错误码 成功-0 失败- errcode

7.8. 轨迹预处理

原型	`LoadTrajectoryJ(name,ovl,opt=1)`

描述	轨迹预处理
必选参数	• `name`:轨迹名,如：/fruser/traj/trajHelix_aima_1.txt; • `ovl`：速度缩放百分比，范围[0~100];
默认参数	• `opt`：1–控制点，默认为1
返回值	错误码 成功–0 失败– errcode

7.9. 轨迹复现

原型	`MoveTrajectoryJ()`
描述	轨迹复现
必选参数	无
默认参数	无
返回值	错误码 成功–0 失败– errcode

7.10. 获取轨迹起始位姿

原型	`GetTrajectoryStartPose(name)`
描述	获取轨迹起始位姿
必选参数	`name`:轨迹名
默认参数	无
返回值	• 错误码 成功–0 失败– errcode • `desc_pose=[x,y,z,rx,ry,rz]`：轨迹起始位姿

7.11. 获取轨迹点编号

原型	`GetTrajectoryPointNum()`
描述	获取轨迹点编号
必选参数	无
默认参数	无
返回值	• 错误码 成功–0 失败– errcode • `pnum`：轨迹点编号

7.12. 设置轨迹运行中的速度

原型	`SetTrajectoryJSpeed(ovl)`
描述	设置轨迹运行中的速度
必选参数	`ovl`:速度缩放百分比，范围[0~100]
默认参数	无
返回值	错误码 成功-0 失败- errcode

7.13. 设置轨迹运行中的力和扭矩

原型	`SetTrajectoryJForceTorque(ft)`
描述	设置轨迹运行中的力和扭矩
必选参数	`ft=[fx,fy,fz,tx,ty,tz]`:单位N和Nm
默认参数	无
返回值	错误码 成功-0 失败- errcode

7.14. 设置轨迹运行中的沿x方向的力

原型	`SetTrajectoryJForceFx(fx)`
描述	设置轨迹运行中的沿x方向的力
必选参数	`ft`:沿x方向的力，单位N
默认参数	无
返回值	错误码 成功-0 失败- errcode

7.15. 设置轨迹运行中的沿y方向的力

原型	`SetTrajectoryJForceFx(fy)`
描述	设置轨迹运行中的沿y方向的力
必选参数	`fy`:沿y方向的力，单位N
默认参数	无
返回值	错误码 成功-0 失败- errcode

7.16. 设置轨迹运行中的沿z方向的力

原型	`SetTrajectoryJForceFx(fz)`
描述	设置轨迹运行中的沿z方向的力
必选参数	`fz`:沿z方向的力，单位N
默认参数	无
返回值	错误码 成功–0 失败– errcode

7.17. 设置轨迹运行中的绕x轴的扭矩

原型	`SetTrajectoryJTorqueTx(tx)`
描述	设置轨迹运行中的绕x轴的扭矩
必选参数	`tx`:绕x轴的扭矩，单位Nm
默认参数	无
返回值	错误码 成功–0 失败– errcode

7.18. 设置轨迹运行中的绕y轴的扭矩

原型	`SetTrajectoryJTorqueTx(ty)`
描述	设置轨迹运行中的绕y轴的扭矩
必选参数	`ty`:绕y轴的扭矩，单位Nm
默认参数	无
返回值	错误码 成功–0 失败– errcode

7.19. 设置轨迹运行中的绕z轴的扭矩

在 Python 版本发生变更: SDK–v2.0.8–3.7.8

原型	`SetTrajectoryJTorqueTx(tz)`
描述	设置轨迹运行中的绕z轴的扭矩
必选参数	• `tz`:绕z轴的扭矩，单位Nm
默认参数	无
返回值	错误码 成功–0 失败– errcode

7.19.1. 代码示例

```python
from fairino import Robot
import time
# 与机器人控制器建立连接，连接成功返回一个机器人对象
robot = Robot.RPC('192.168.58.2')
name = "/fruser/traj/trajHelix_aima_1.txt"    #轨迹名
blend = 1   #是否平滑，1-平滑，0-不平滑
ovl = 50.0   #速度缩放
ft =[0.5, 0.5, 0.5, 0.5, 0.5, 0.5]
ret = robot.LoadTrajectoryJ(name,ovl)  #轨迹预加载
print("轨迹预加载错误码",ret)
ret,P1 = robot.GetTrajectoryStartPose(name)    #获取轨迹起始位姿
print ("获取轨迹起始位姿错误码",ret,"起始位姿",P1)
ret = robot.MoveL(P1,1,0)        #运动到起始点
print("运动到起始点错误码",ret)
ret = robot.GetTrajectoryPointNum()        #获取轨迹点编号
print("获取轨迹点编号错误码",ret)
time.sleep(10)
ret = robot.MoveTrajectoryJ()  #轨迹复现
print("轨迹复现错误码",ret)
time.sleep(10)
ret = robot.SetTrajectoryJSpeed(ovl)  #设置轨迹运行中的速度
print("设置轨迹运行中的速度错误码",ret)
time.sleep(1)
ret = robot.SetTrajectoryJForceTorque(ft)  #设置轨迹运行中的力和扭矩
print("设置轨迹运行中的力和扭矩错误码",ret)
time.sleep(1)
ret = robot.SetTrajectoryJForceFx(0) #设置轨迹运行中的沿x方向的力
print("设置轨迹运行中的沿x方向的力错误码",ret)
time.sleep(1)
ret = robot.SetTrajectoryJForceFy(0) #设置轨迹运行中的沿y方向的力
print("设置轨迹运行中的沿y方向的力错误码",ret)
time.sleep(1)
ret = robot.SetTrajectoryJForceFz(0) #设置轨迹运行中的沿z方向的力
print("设置轨迹运行中的沿z方向的力错误码",ret)
time.sleep(1)
ret = robot.SetTrajectoryJTorqueTx(0) #设置轨迹运行中的绕x轴的扭矩
print("设置轨迹运行中的绕x轴的扭矩错误码",ret)
time.sleep(1)
ret = robot.SetTrajectoryJTorqueTy(0) #设置轨迹运行中的绕y轴的扭矩
print("设置轨迹运行中的绕y轴的扭矩错误码",ret)
time.sleep(1)
ret = robot.SetTrajectoryJTorqueTz(0) #设置轨迹运行中的绕z轴的扭矩
print("设置轨迹运行中的绕z轴的扭矩错误码",ret)
time.sleep(1)
```

7.20. 上传轨迹J文件

在 Python 版本加入: SDK-v2.0.8-3.7.8

原型	`TrajectoryJUpLoad(filePath)`
描述	上传轨迹J文件
必选参数	• `filePath`:上传轨迹文件的全路径名，C://test/tes...

默认参数	无
返回值	错误码 成功–0 失败– errcode

7.21. 删除轨迹J文件

在 Python 版本加入: SDK–v2.0.8–3.7.8

原型	`TrajectoryJDelete(filePath)`
描述	删除轨迹J文件
必选参数	• `filePath`:删除轨迹文件的全路径名，C://test/testJ.txt
默认参数	无
返回值	错误码 成功–0 失败– errcode

7.21.1. 代码示例

```
from fairino import Robot
import time
# 与机器人控制器建立连接，连接成功返回一个机器人对象
robot = Robot.RPC('192.168.58.2')
robot.LoggerInit()
robot.SetLoggerLevel(lvl=1)

retval = robot.TrajectoryJDelete("testA.txt")
print("TrajectoryJDelete return ", retval)
robot.TrajectoryJUpLoad("D://zUP/testA.txt")

traj_file_name = "/fruser/traj/testA.txt"
retval = robot.LoadTrajectoryJ(traj_file_name, 100, 1)
print("LoadTrajectoryJ return ", retval)

retval,traj_start_pose = robot.GetTrajectoryStartPose(traj_file_name)
print("GetTrajectoryStartPose return ", retval)
print("轨迹起始位姿:", traj_start_pose[0], traj_start_pose[1], traj_start_pose[2], traj_start_pose[3], traj_start_pose[4], traj_start_pose[5])

robot.SetSpeed(20)
robot.MoveCart(traj_start_pose, 1, 0)

time.sleep(5)

retval,traj_num = robot.GetTrajectoryPointNum()
print("GetTrajectoryPointNum return ", retval)
print("轨迹点编号: ", traj_num)

retval = robot.MoveTrajectoryJ()
print("MoveTrajectoryJ return ", retval)
```

7.22. 轨迹预处理(轨迹前瞻)

在 python 版本加入: SDK-v2.1.0

原型	`LoadTrajectoryLA(name, mode, errorLim, type, precision, vamx, amax, jmax)`
描述	轨迹预处理(轨迹前瞻)
必选参数	• `name`:轨迹文件名 • `mode`：采样模式，0-不进行采样；1-等数据间隔采样；2-等误差限制采样 • `errorLim`:误差限制，使用直线拟合生效 • `type`:平滑方式，0-贝塞尔平滑 • `precision`:平滑精度，使用贝塞尔平滑时生效 • `vamx`:设定的最大速度，mm/s • `amax`:设定的最大加速度，mm/s2 • `jmax`:设定的最大加加速度，mm/s3
默认参数	无
返回值	错误码 成功-0 失败- errcode

7.23. 轨迹复现(轨迹前瞻)

在 python 版本加入: SDK-v2.1.0

原型	`MoveTrajectoryLA()`
描述	轨迹复现(轨迹前瞻)
必选参数	无
默认参数	无
返回值	错误码 成功-0 失败- errcode

7.23.1. 代码示例

```python
from fairino import Robot
import time
# 与机器人控制器建立连接，连接成功返回一个机器人对象
robot = Robot.RPC('192.168.58.2')

rtn = 0
rtn = robot.TrajectoryJUpLoad("D://zUP/A.txt")
print("TrajectoryJUpLoad A.txt rtn is ",rtn)
rtn = robot.TrajectoryJUpLoad("D://zUP/B.txt")
print("TrajectoryJUpLoad B.txt rtn is ", rtn)
nameA = "/fruser/traj/A.txt"
nameB = "/fruser/traj/B.txt"

# rtn = robot.LoadTrajectoryLA(nameA, 2, 0.0, 0, 1.0, 100.0, 200.0, 1000.0) #B样条
# print("LoadTrajectoryLA rtn is ", rtn)
robot.LoadTrajectoryLA(nameB, 0, 0, 0, 1, 100, 100, 1000) #直线连接
# robot.LoadTrajectoryLA(nameA, 1, 2, 0, 2, 100, 200, 1000) #直线拟合
# error,startPos = robot.GetTrajectoryStartPose(nameA)
error,startPos = robot.GetTrajectoryStartPose(nameB)
robot.MoveCart(startPos, 1, 0, 100, 100, 100, -1, -1)
rtn = robot.MoveTrajectoryLA()
print("MoveTrajectoryLA rtn is ", rtn)
```