10. 机器人力控

10.1. 获取力传感器配置

17-坤维科技，19-航天十一院，20-ATI 传感器，21-中科米点，22-伟航敏芯; - device 设备号，坤维 (0-KWI

10.1.1. 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')
company = 17    #传感器厂商，17-坤维科技
device = 0      #传感器设备号
error = robot.FT_SetConfig(company, device)   #配置力传感器
print("配置力传感器错误码",error)
config = robot.FT_GetConfig() #获取力传感器配置信息
print('获取力传感器配置信息',config)
time.sleep(1)
error = robot.FT_Activate(0)  #传感器复位
print("传感器复位错误码",error)
time.sleep(1)
error = robot.FT_Activate(1)  #传感器激活
print("传感器激活错误码",error)
time.sleep(1)
error = robot.SetLoadWeight(0.0)    #末端负载设置为零
print("末端负载设置为零错误码",error)
time.sleep(1)
error = robot.SetLoadCoord(0.0,0.0,0.0)  #末端负载质心设置为零
print("末端质心设置为零错误码",error)
time.sleep(1)
error = robot.FT_SetZero(0)   #传感器去除零点
print("传感器去除零点错误码",error)
time.sleep(1)
error = robot.FT_GetForceTorqueOrigin()   #查询传感器原始数据
print("查询传感器原始数据",error)
error = robot.FT_SetZero(1)   #传感器零点矫正,注意此时末端不能安装工具，只有力传感器
print("传感器零点矫正",error)
time.sleep(1)
error = robot.FT_GetForceTorqueRCS()  #查询传感器坐标系下数据
print("查询传感器坐标系下数据",error)
```

10.2. 力传感器配置

:company,device,softversion=0,bus=0)

传感器厂商，17-坤维科技，19-航天十一院，20-ATI传感器，21-中科米点，22-伟航敏芯；
备号，坤维(0-KWR75B)，航天十一院(0-MCS6A-200-4)，ATI(0-AXIA80-M8)，中科米点(0-MST2010)，伟

：软件版本号，暂不使用，默认为0；
挂载末端总线位置，暂不使用，默认为 0；

10.3. 力传感器激活

原型	FT_Activate(state)
描述	力传感器激活
必选参数	• `state`：0-复位，1-激活
默认参数	无
返回值	错误码 成功-0 失败- errcode

10.4. 力传感器校零

原型	FT_SetZero(state)
描述	力传感器校零
必选参数	• `state`：0-去除零点，1-零点矫正
默认参数	无
返回值	错误码 成功-0 失败- errcode

10.5. 设置力传感器参考坐标系

在 python 版本发生变更: SDK-v2.0.5

原型	FT_SetRCS(ref)
描述	设置力传感器参考坐标系
必选参数	• `ref`：0-工具坐标系，1-基坐标系
默认参数	• `coord`：[x,y,z,rx,ry,rz]自定义坐标系值,默认[0,0,0,0,0,0]
返回值	错误码 成功-0 失败- errcode

10.5.1. 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')
#负载辨识，此时末端安装要辨识的工具，工具安装在力传感器下方,末端竖直向下
error = robot.FT_SetRCS(0)    #设置参考坐标系为工具坐标系，0-工具坐标系，1-基坐标系
print('设置参考坐标系错误码',error)
time.sleep(1)
tool_id = 10  #传感器坐标系编号
tool_coord = [0.0,0.0,35.0,0.0,0.0,0.0]   # 传感器相对末端法兰位姿
tool_type = 1  # 0-工具，1-传感器
tool_install = 0 # 0-安装末端，1-机器人外部
error = robot.SetToolCoord(tool_id,tool_coord,tool_type,tool_install)     #设置传感器坐标系，传感器相对末端法兰位姿
print('设置传感器坐标系错误码',error)
time.sleep(1)
error = robot.FT_PdIdenRecord(tool_id)   #记录辨识数据
print('记录负载重量错误码',error)
time.sleep(1)
error = robot.FT_PdIdenRecord()  #计算负载重量，单位kg
print('计算负载重量错误码',error)
#负载质心辨识，机器人需要示教三个不同的姿态，然后记录辨识数据，最后计算负载质心
robot.Mode(1)
ret = robot.DragTeachSwitch(1)  #机器人切入拖动示教模式，必须在手动模式下才能切入拖动示教模式
time.sleep(5)
ret = robot.DragTeachSwitch(0)
time.sleep(1)
error = robot.FT_PdCogIdenRecord(tool_id,1)
print('负载质心1错误码',error)#记录辨识数据
ret = robot.DragTeachSwitch(1)  #机器人切入拖动示教模式，必须在手动模式下才能切入拖动示教模式
time.sleep(5)
ret = robot.DragTeachSwitch(0)
time.sleep(1)
error = robot.FT_PdCogIdenRecord(tool_id,2)
print('负载质心2错误码',error)
ret = robot.DragTeachSwitch(1)  #机器人切入拖动示教模式，必须在手动模式下才能切入拖动示教模式
time.sleep(5)
ret = robot.DragTeachSwitch(0)
time.sleep(1)
error = robot.FT_PdCogIdenRecord(tool_id,3)
print('负载质心3错误码',error)
time.sleep(1)
error = robot.FT_PdCogIdenCompute()
print('负载质心计算错误码',error)
```

10.6. 负载重量辨识计算

原型	FT_PdIdenCompute()
描述	负载重量辨识计算
必选参数	无
默认参数	无
返回值	错误码 成功-0 失败- errcode - 返回值（调用成功返回）weight-负载重量，单位 kg

10.7. 负载重量辨识记录

原型	FT_PdIdenRecord(tool_id)
描述	负载重量辨识记录

必选参数	• `tool_id`：传感器坐标系编号，范围[0~14]
默认参数	无
返回值	错误码 成功-0 失败- errcode

10.8. 负载质心辨识计算

型	FT_PdCogIdenCompute ()
述	负载质心辨识计算
参数	无
参数	无
回值	错误码 成功-0 失败- errcode - 返回值（调用成功返回）cog=[cogx,cogy,cogz]，负载质心，单位

10.9. 负载质心辨识记录

原型	FT_PdCogIdenRecord(tool_id,index)
描述	负载质心辨识记录
必选参数	• `tool_id`：传感器坐标系编号，范围[0~14]; • `index`：点编号[1~3]
默认参数	无
返回值	错误码 成功-0 失败- errcode

10.10. 获取参考坐标系下力/扭矩数据

原型	FT_GetForceTorqueRCS()
描述	获取参考坐标系下力/扭矩数据
必选参数	无
默认参数	无
返回值	错误码 成功-0 失败- errcode - 返回值（调用成功返回）data=[fx,fy,fz,tx,ty,tz]

10.10.1. 代码示例

```python
import frrpc
# 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 = frrpc.RPC('192.168.58.2')
rcs = robot.FT_GetForceTorqueRCS()  #查询传感器坐标系下数据
print(rcs)
```

10.11. 获取力传感器原始力/扭矩数据

原型	FT_GetForceTorqueOrigin()

描述	获取力传感器原始力/扭矩数据
必选参数	无
默认参数	无
返回值	错误码 成功-0 失败- errcode - 返回值（调用成功返回）data=[fx,fy,fz,tx,ty,tz]

10.11.1. 代码示例

```python
import frrpc
# 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 = frrpc.RPC('192.168.58.2')
origin = robot.FT_GetForceTorqueOrigin()    #查询传感器原始数据
print(origin)
```

10.12. 碰撞守护

原型	FT_Guard(flag,sensor_num,select,force_torque,max_threshold,min_threshold)
描述	碰撞守护
必选参数	• `flag`：0-关闭碰撞守护，1-开启碰撞守护； • `sensor_num`：力传感器编号； • `select`：六个自由度是否检测碰撞[fx,fy,fz,mx,my,mz]，0-不生效，1-生效； • `force_torque`：碰撞检测力/力矩，单位N或Nm； • `max_threshold`：最大阈值； • `min_threshold`：最小阈值； • 力/力矩检测范围:(force_torque-min_threshold,force_torque+max_threshold)
默认参数	无
返回值	错误码 成功-0 失败- errcode

10.12.1. 代码示例

```
from fairino import Robot
# 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')
#碰撞守护
actFlag = 1   #开启标志，0-关闭碰撞守护，1-开启碰撞守护
sensor_num = 1  #力传感器编号
is_select = [1,1,1,1,1,1]  #六个自由度选择[fx,fy,fz,mx,my,mz]，0-不生效，1-生效
force_torque = [0.0,0.0,0.0,0.0,0.0,0.0]  #碰撞检测力和力矩，检测范围（force_torque-min_threshold,force_torque+max_threshold）
max_threshold = [10.0,10.0,10.0,10.0,10.0,10.0]  #最大阈值
min_threshold = [5.0,5.0,5.0,5.0,5.0,5.0]   #最小阈值
P1=[-160.619,-586.138,384.988,-170.166,-44.782,169.295]
P2=[-87.615,-606.209,556.119,-102.495,10.118,178.985]
P3=[41.479,-557.243,484.407,-125.174,46.995,-132.165]
error = robot.FT_Guard(actFlag, sensor_num, is_select, force_torque, max_threshold, min_threshold)    #开启碰撞守护
print("开启碰撞守护错误码",error)
error = robot.MoveL(P1,1,0)          #笛卡尔空间直线运动
print("笛卡尔空间直线运动错误码",error)
error = robot.MoveL(P2,1,0)
print("笛卡尔空间直线运动错误码",error)
error = robot.MoveL(P3,1,0)
print("笛卡尔空间直线运动错误码",error)
actFlag = 0
error = robot.FT_Guard(actFlag, sensor_num, is_select, force_torque, max_threshold, min_threshold)    #关闭碰撞守护
print("关闭碰撞守护错误码",error)
```

10.13. 恒力控制

原型	FT_Control(flag,sensor_num,select,force_torque,gain,adj_sign,ILC_sign,max_dis,max_ang)
描述	恒力控制
必选参数	• `flag`：恒力控制开启标志，0-关，1-开； • `sensor_num`：力传感器编号； • `select`：六个自由度是否检测 [fx,fy,fz,mx,my,mz]，0-不生效，1-生效； • `force_torque`：检测力/力矩，单位N或Nm； • `gain`：[f_p,f_i,f_d,m_p,m_i,m_d],力PID参数，力矩PID参数； • `adj_sign`：自适应启停状态，0-关闭，1-开启； • `ILC_sign`：ILC控制启停状态，0-停止，1-训练，2-实操； • `max_dis`：最大调整距离； • `max_ang`：最大调整角度；
默认参数	无
返回值	• 成功：[0] • 失败：[errcode]

10.13.1. 代码示例

```
from fairino import Robot
# 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')
#恒力控制
status = 1  #恒力控制开启标志，0-关，1-开
sensor_num = 1 #力传感器编号
is_select = [0,0,1,0,0,0]  #六个自由度选择[fx,fy,fz,mx,my,mz]，0-不生效，1-生效
force_torque = [0.0,0.0,-10.0,0.0,0.0,0.0]
gain = [0.0005,0.0,0.0,0.0,0.0,0.0]  #力PID参数，力矩PID参数
adj_sign = 0  #自适应启停状态，0-关闭，1-开启
ILC_sign = 0  #ILC控制启停状态，0-停止，1-训练，2-实操
max_dis = 100.0  #最大调整距离
max_ang = 0.0  #最大调整角度
J1=[70.395, -46.976, 90.712, -133.442, -87.076, -27.138]
P2=[-123.978, -674.129, 44.308, -178.921, 2.734, -172.449]
P3=[123.978, -674.129, 42.308, -178.921, 2.734, -172.449]
error = robot.MoveJ(J1,1,0)
print("关节空间运动指令错误码",error)
error = robot.MoveL(P2,1,0)
print("笛卡尔空间直线运动指令错误码",error)
error = robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制开启错误码",error)
error = robot.MoveL(P3,1,0)   #笛卡尔空间直线运动
print("笛卡尔空间直线运动指令错误码",error)
status = 0
error = robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制结束错误码",error)
```

10.14. 螺旋线探索

原型	FT_SpiralSearch(rcs,ft, dr = 0.7,max_t_ms = 60000, max_vel = 5)
描述	螺旋线探索
必选参数	• `rcs`：参考坐标系，0-工具坐标系，1-基坐标系 • `ft`：力或力矩阈值 (0~100)，单位 N 或 Nm;
默认参数	• `dr`：每圈半径进给量，单位 mm 默认0.7; • `max_t_ms`：最大探索时间，单位 ms 默认 60000; • `max_vel`：线速度最大值，单位 mm/s 默认 5
返回值	• 成功：[0] • 失败：[errcode]

10.14.1. 代码示例

```
from fairino import Robot
# 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')
P = [36.794,-675.119, 65.379, -176.938, 2.535, -179.829]
#恒力参数
status = 1  #恒力控制开启标志，0-关，1-开
sensor_num = 1 #力传感器编号
is_select = [0,0,1,0,0,0]  #六个自由度选择[fx,fy,fz,mx,my,mz]，0-不生效，1-生效
force_torque = [0.0,0.0,-10.0,0.0,0.0,0.0]
gain = [0.0001,0.0,0.0,0.0,0.0,0.0]  #力PID参数，力矩PID参数
adj_sign = 0  #自适应启停状态，0-关闭，1-开启
ILC_sign = 0  #ILC控制启停状态，0-停止，1-训练，2-实操
max_dis = 100.0  #最大调整距离
max_ang = 5.0  #最大调整角度
#螺旋线探索参数
rcs = 0  #参考坐标系，0-工具坐标系，1-基坐标系
fFinish = 10 #力或力矩阈值（0~100），单位N或Nm
error = robot.MoveL(P,1,0) #笛卡尔空间直线运动至初始点
print("笛卡尔空间直线运动至初始点",error)
is_select = [0,0,1,1,1,0]  #六个自由度选择[fx,fy,fz,mx,my,mz]，0-不生效，1-生效
error = robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign, ILC_sign,
max_dis,max_ang)
print("恒力控制开启错误码",error)
error = robot.FT_SpiralSearch(rcs,fFinish,max_vel=3)
print("螺旋线探索错误码",error)
status = 0
error = robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign, ILC_sign,
max_dis,max_ang)
print("恒力控制关闭错误码",error)
```

10.15. 旋转插入

原型	FT_RotInsertion(rcs,ft, orn, angVelRot = 3, angleMax = 45, angAccmax = 0, rotorn =1)
描述	旋转插入
必选参数	• `rcs`：参考坐标系，0-工具坐标系，1-基坐标系； • `ft`：力或力矩阈值 (0~100)，单位 N 或 Nm; • `orn`：力/扭矩方向，1-沿z轴方向，2-绕z轴方向;
默认参数	• `angVelRot`：旋转角速度，单位°/s,默认 3; • `angleMax`：最大旋转角度，单位 ° 默认 5; • `angAccmax`：最大旋转加速度，单位 °/s^2，暂不使用 默认0; • `rotorn`：旋转方向，1-顺时针，2-逆时针 默认1
返回值	• 成功：[0] • 失败：[errcode]

10.15.1. 代码示例

```
from fairino import Robot
# 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')
P = [36.794,-675.119, 65.379, -176.938, 2.535, -179.829]
#恒力参数
status = 1  #恒力控制开启标志，0-关，1-开
sensor_num = 1 #力传感器编号
is_select = [0,0,1,0,0,0]  #六个自由度选择[fx,fy,fz,mx,my,mz]，0-不生效，1-生效
force_torque = [0.0,0.0,-10.0,0.0,0.0,0.0]
gain = [0.0001,0.0,0.0,0.0,0.0,0.0]  #力PID参数，力矩PID参数
adj_sign = 0  #自适应启停状态，0-关闭，1-开启
ILC_sign = 0  #ILC控制启停状态，0-停止，1-训练，2-实操
max_dis = 100.0  #最大调整距离
max_ang = 5.0  #最大调整角度
#旋转插入参数
rcs = 0  #参考坐标系，0-工具坐标系，1-基坐标系
forceInsertion = 2.0 #力或力矩阈值（0~100），单位N或Nm
orn = 1 #力的方向，1-fz,2-mz
#默认参数 angVelRot：旋转角速度，单位 °/s  默认 3
#默认参数 angleMax：最大旋转角度，单位 ° 默认 5
#默认参数 angAccmax：最大旋转加速度，单位 °/s^2，暂不使用 默认0
#默认参数 rotorn：旋转方向，1-顺时针，2-逆时针 默认1
error = robot.MoveL(P,1,0) #笛卡尔空间直线运动至初始点
print("笛卡尔空间直线运动至初始点",error)
error = robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制开启错误码",error)
error = robot.FT_RotInsertion(rcs,1,orn)
print("旋转插入错误码",error)
status = 0
error = robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制关闭错误码",error)
```

10.16. 直线插入

原型	FT_LinInsertion(rcs, ft, disMax, linorn, lin_v = 1.0, lin_a = 1.0)
描述	直线插入
必选参数	• `rcs`：参考坐标系，0-工具坐标系，1-基坐标系； • `ft`：力或力矩阈值 (0~100)，单位 N 或 Nm; • `disMax`：最大插入距离，单位 mm; • `linorn`：插入方向:0-负方向，1-正方向
默认参数	• `lin_v`：直线速度，单位 mm/s 默认1; • `lin_a`：直线加速度，单位 mm/s^2，暂不使用 默认0
返回值	• 成功：[0] • 失败：[errcode]

10.16.1. 代码示例

```
from fairino import Robot
# 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')
P = [36.794,-675.119, 65.379, -176.938, 2.535, -179.829]
#恒力参数
status = 1  #恒力控制开启标志，0-关，1-开
sensor_num = 1 #力传感器编号
is_select = [0,0,1,0,0,0]  #六个自由度选择[fx,fy,fz,mx,my,mz]，0-不生效，1-生效
force_torque = [0.0,0.0,-10.0,0.0,0.0,0.0]
gain = [0.0001,0.0,0.0,0.0,0.0,0.0]  #力PID参数，力矩PID参数
adj_sign = 0  #自适应启停状态，0-关闭，1-开启
ILC_sign = 0  #ILC控制启停状态，0-停止，1-训练，2-实操
max_dis = 100.0  #最大调整距离
max_ang = 5.0  #最大调整角度
#直线插入参数
rcs = 0  #参考坐标系，0-工具坐标系，1-基坐标系
force_goal = 10.0  #力或力矩阈值（0~100），单位N或Nm
disMax = 100.0 #最大插入距离，单位mm
linorn = 1 #插入方向，1-正方向，2-负方向
#默认参数 lin_v：直线速度，单位 mm/s 默认1
#默认参数 lin_a：直线加速度，单位 mm/s^2，暂不使用 默认0
error = robot.MoveL(P,1,0) #笛卡尔空间直线运动至初始点
print("笛卡尔空间直线运动至初始点",error)
error = robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制开启错误码",error)
error = robot.FT_LinInsertion(rcs,force_goal,disMax,linorn)
print("直线插入错误码",error)
status = 0
error = robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制关闭错误码",error)
```

10.17. 计算中间平面位置开始

原型	FT_CalCenterStart()
描述	计算中间平面位置开始
必选参数	无
默认参数	无
返回值	错误码 成功-0 失败- errcode

10.18. 计算中间平面位置结束

原型	FT_CalCenterEnd()
描述	计算中间平面位置结束
必选参数	无
默认参数	无
返回值	错误码 成功-0 失败- errcode - 返回值（调用成功返回）pos=[x,y,z,rx,ry,rz]

10.19. 表面定位

原型	FT_FindSurface (rcs, dir, axis, disMax, ft, lin_v = 3.0, lin_a = 0.0)
描述	表面定位
必选参数	• `rcs`： 参考坐标系，0-工具坐标系，1-基坐标系； • `dir`：移动方向，1-正方向，2-负方向； • `axis`：移动轴，1-x，2-y，3-z； • `disMax`：大探索距离，单位 mm; • `ft`：动作终止力阈值，单位N；
默认参数	• `lin_v`：探索直线速度，单位mm/s 默认3; • `lin_a`：探索直线加速度，单位mm/s^2 默认0;
返回值	错误码 成功-0 失败- errcode

10.19.1. 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')
#恒力控制
status = 1  #恒力控制开启标志，0-关，1-开
sensor_num = 1 #力传感器编号
is_select = [1,0,0,0,0,0]  #六个自由度选择[fx,fy,fz,mx,my,mz]，0-不生效，1-生效
force_torque = [-2.0,0.0,0.0,0.0,0.0,0.0]
gain = [0.0002,0.0,0.0,0.0,0.0,0.0]  #力PID参数，力矩PID参数
adj_sign = 0  #自适应启停状态，0-关闭，1-开启
ILC_sign = 0  #ILC控制启停状态，0-停止，1-训练，2-实操
max_dis = 15.0  #最大调整距离
max_ang = 0.0  #最大调整角度
#表面定位参数
rcs = 0 #参考坐标系，0-工具坐标系，1-基坐标系
direction = 1 #移动方向，1-正方向，2-负方向
axis = 1 #移动轴，1-X,2-Y,3-Z
lin_v = 3.0  #探索直线速度，单位mm/s
lin_a = 0.0  #探索直线加速度，单位mm/s^2
disMax = 50.0 #最大探索距离，单位mm
force_goal = 2.0 #动作终止力阈值，单位N
P1=[-77.24,-679.599,58.328,179.373,-0.028,-167.849]
Robot.MoveCart(P1,1,0)       #关节空间点到点运动
#x方向寻找中心
#第1个表面
error = robot.FT_CalCenterStart()
print("计算中间平面开始错误码",error)
error = robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制开始错误码",error)
error = robot.FT_FindSurface(rcs,direction,axis,disMax,force_goal)
print("寻找X+表面错误码",error)
status = 0
error = robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制结束错误码",error)
time.sleep(2)
error = robot.MoveCart(P1,1,0)       #关节空间点到点运动
print("关节空间点到点运动错误码",error)
time.sleep(5)
#第2个表面
error = robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制开始错误码",error)
direction = 2 #移动方向，1-正方向，2-负方向
error = robot.FT_FindSurface(rcs,direction,axis,disMax,force_goal)
print("寻找X-表面错误码",error)
status = 0
error = robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制结束错误码",error)
#计算x方向中心位置
error,xcenter = robot.FT_CalCenterEnd()
print("计算X方向中间平面结束错误码",xcenter)
error = robot.MoveCart(xcenter,1,0)
print("关节空间点到点运动错误码",error)
time.sleep(1)
#y方向寻找中心
#第1个表面
error =robot.FT_CalCenterStart()
print("计算中间平面开始错误码",error)
error =robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制开始错误码",error)
direction = 1 #移动方向，1-正方向，2-负方向
axis = 2 #移动轴，1-X,2-Y,3-Z
disMax = 150.0 #最大探索距离，单位mm
```

```
lin_v = 6.0  #探索直线速度，单位mm/s
error =robot.FT_FindSurface(rcs,direction,axis,disMax,force_goal)
print("寻找表面Y+错误码",error)
status = 0
error =robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制结束错误码",error)
error =robot.MoveCart(P1,1,0)       #关节空间点到点运动
print("关节空间点到点运动错误码",error)
Robot.WaitMs(1000)
#第2个表面
error =robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制开始错误码",error)
direction = 2 #移动方向，1-正方向，2-负方向
error =robot.FT_FindSurface(rcs,direction,axis,disMax,force_goal)
print("寻找表面Y-错误码",error)
status = 0
error =robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制结束错误码",error)
#计算y方向中心位置
error,ycenter=robot.FT_CalCenterEnd()
print("计算中间平面Y方向结束错误码",ycenter)
error =robot.MoveCart(ycenter,1,0)
print("关节空间点到点运动错误码",error)
```

10.20. 柔顺控制关闭

原型	FT_ComplianceStop()
描述	柔顺控制关闭
必选参数	无
默认参数	无
返回值	错误码 成功-0 失败- errcode

10.21. 柔顺控制开启

原型	FT_ComplianceStart(p,force)
描述	柔顺控制开启
必选参数	• `p`：位置调节系数或柔顺系数 • `force`：柔顺开启力阈值，单位N
默认参数	无
返回值	错误码 成功-0 失败- errcode

10.21.1. 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')
J1=[75.005,-46.434,90.687,-133.708,-90.315,-27.139]
P2=[-77.24,-679.599,38.328,179.373,-0.028,-167.849]
P3=[77.24,-679.599,38.328,179.373,-0.028,-167.849]
#恒力控制参数
status = 1  #恒力控制开启标志，0-关，1-开
sensor_num = 1 #力传感器编号
is_select = [1,1,1,0,0,0]  #六个自由度选择[fx,fy,fz,mx,my,mz]，0-不生效，1-生效
force_torque = [-10.0,-10.0,-10.0,0.0,0.0,0.0]
gain = [0.0005,0.0,0.0,0.0,0.0,0.0]  #力PID参数，力矩PID参数
adj_sign = 0  #自适应启停状态，0-关闭，1-开启
ILC_sign = 0  #ILC控制启停状态，0-停止，1-训练，2-实操
max_dis = 1000.0  #最大调整距离
max_ang = 0.0  #最大调整角度
error = robot.MoveJ(J1,1,0)
print("关节空间运动到点1错误码",error)
#柔顺控制
error = robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制开始错误码",error)
p = 0.00005  #位置调节系数或柔顺系数
force = 30.0 #柔顺开启力阈值，单位N
error = robot.FT_ComplianceStart(p,force)
print("柔顺控制开始错误码",error)
error = robot.MoveL(P2,1,0,vel =10)   #笛卡尔空间直线运动
print("笛卡尔空间直线运动到点2错误码", error)
error = robot.MoveL(P3,1,0,vel =10)
print("笛卡尔空间直线运动到点3错误码", error)
time.sleep(1)
error = robot.FT_ComplianceStop()
print("柔顺控制结束错误码",error)
status = 0
error = robot.FT_Control(status,sensor_num,is_select,force_torque,gain,adj_sign,
ILC_sign,max_dis,max_ang)
print("恒力控制关闭错误码",error)
```

10.22. 负载辨识滤波初始化

在 python 版本加入: SDK-v2.0.1

原型	LoadIdentifyDynFilterInit()
描述	负载辨识滤波初始化
必选参数	无
默认参数	无
返回值	错误码 成功-0 失败- errcode

10.22.1. 代码示例

```python
from fairino import Robot

# 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')

#负载辨识滤波初始化
error = robot.LoadIdentifyDynFilterInit()
print("LoadIdentifyDynFilterInit:",error)
#负载辨识变量初始化
error = robot.LoadIdentifyDynVarInit()
print("LoadIdentifyDynVarInit:",error)

joint_torque= [0,0,0,0,0,0]
joint_pos= [0,0,0,0,0,0]
gain=[0,0.05,0,0,0,0,0,0.02,0,0,0,0]
t =10
error,joint_pos=robot.GetActualJointPosDegree(1)
joint_pos[1] = joint_pos[1]+10
error,joint_torque=robot.GetJointTorques(1)
joint_torque[1] = joint_torque[1]+2
#负载辨识主程序
error = robot.LoadIdentifyMain(joint_torque, joint_pos, t)
print("LoadIdentifyMain:",error)
#获取负载辨识结果
error = robot.LoadIdentifyGetResult(gain)
print("LoadIdentifyGetResult:",error)
```

10.23. 负载辨识变量初始化

在 python 版本加入: SDK-v2.0.1

原型	LoadIdentifyDynVarInit()
描述	负载辨识变量初始化
必选参数	无
默认参数	无
返回值	错误码 成功-0 失败- errcode

10.24. 负载辨识主程序

在 python 版本加入: SDK-v2.0.1

原型	LoadIdentifyMain(joint_torque, joint_pos, t)
描述	负载辨识主程序
必选参数	• `joint_torque`： 关节扭矩 j1-j6； • `joint_pos`：关节位置 j1-j6
默认参数	无
返回值	错误码 成功-0 失败- errcode

10.25. 获取负载辨识结果

在 python 版本加入: SDK-v2.0.1

原型	LoadIdentifyGetResult(gain)
描述	获取负载辨识结果
必选参数	• `gain`
默认参数	无
返回值	• 错误码 成功-0 失败- errcode • Return:（if success）weight 负载重量，cog 负载质心 [x,y,z]

10.26. 力传感器辅助拖动

在 python 版本加入: SDK-v2.0.5

rceAndJointImpedanceStartStop(status, impedanceFlag, lamdeDain, KGain, BGain, dragMaxTcpVel,dra

专感器辅助拖动

`status`：控制状态，0-关闭；1-开启
`impedanceFlag`：阻抗开启标志，0-关闭；1-开启
`lamdeDain`：[D1,D2,D3,D4,D5, D6] 拖动增益
`KGain`：[K1,K2,K3,K4,K5,K6]刚度增益
`BGain`：[B1,B2,B3,B4,B5,B]阻尼增益
`dragMaxTcpVel`：拖动末端最大线速度限制
`dragMaxTcpOriVel`：拖动末端最大角速度限制

错误码 成功-0 失败- errcode

10.26.1. 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')

status = 1 #控制状态，0-关闭；1-开启
asaptiveFlag = 1 #自适应开启标志，0-关闭；1-开启
interfereDragFlag = 1 #干涉区拖动标志，0-关闭；1-开启
M = [15, 15, 15, 0.5, 0.5, 0.1] #惯性系数
B = [150, 150, 150, 5, 5, 1] #阻尼系数
K = [0, 0, 0, 0, 0, 0] #刚度系数
F = [5, 5, 5, 1, 1, 1] #拖动六维力阈值
Fmax = 50 #最大拖动力限制
Vmax = 1810 #最大关节速度限制

error = robot.EndForceDragControl(status, asaptiveFlag, interfereDragFlag, M, B, K, F, Fmax, Vmax)
print("EndForceDragControl return:",error)

time.sleep(10)
status=0
error = robot.EndForceDragControl(status, asaptiveFlag, interfereDragFlag, M, B, K, F, Fmax, Vmax)
print("EndForceDragControl return:",error)
```

10.27. 报错清除后力传感器自动开启

在 python 版本加入: SDK-v2.0.5

原型	SetForceSensorDragAutoFlag(status)
描述	报错清除后力传感器自动开启
必选参数	• `status`：控制状态，0-关闭；1-开启
默认参数	无
返回值	• 错误码 成功-0 失败- errcode

10.27.1. 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')

error = robot. SetForceSensorDragAutoFlag (1)
print("SetForceSensorDragAutoFlag return:",error)
```

10.28. 设置六维力和关节阻抗混合拖动开关及参数

在 python 版本加入: SDK-v2.0.5

原型	EndForceDragControl(status, asaptiveFlag, interfereDragFlag, M, B, K, F, Fmax, Vmax)

描述	设置六维力和关节阻抗混合拖动开关及参数
必选参数	• `status`：控制状态，0-关闭；1-开启 • `asaptiveFlag`：自适应开启标志，0-关闭；1-开启 • `interfereDragFlag`：干涉区拖动标志，0-关闭；1-开启 • `M=[m1,m2,m3,m4,m5,m6]`：惯性系数 • `B=[b1,b2,b3,b4,b5,b6]`：阻尼系数 • `K=[k1,k2,k3,k4,k5,k6]`：刚度系数 • `F=[f1,f2,f3,f4,f5,f6]`：拖动六维力阈值 • `Fmax`：最大拖动力限制 • `Vmax`：最大关节速度限制
默认参数	无
返回值	• 错误码 成功-0 失败-errcode

10.28.1. 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')

status = 1 #控制状态，0-关闭；1-开启
impedanceFlag = 1 #阻抗开启标志，0-关闭；1-开启
lamdeDain = [ 3.0, 2.0, 2.0, 2.0, 2.0, 3.0] # 拖动增益
KGain = [0.00, 0.00, 0.00, 0.00, 0.00, 0.00] # 刚度增益
BGain = [150, 150, 150, 5.0, 5.0, 1.0] # 阻尼增益
dragMaxTcpVel = 1000 #拖动末端最大线速度限制
dragMaxTcpOriVel = 180 #拖动末端最大角速度限制

error = robot.DragTeachSwitch(1)
print("DragTeachSwitch 1  return:",error)

error = robot.ForceAndJointImpedanceStartStop(status, impedanceFlag, lamdeDain, KGain,
BGain,dragMaxTcpVel,dragMaxTcpOriVel)
print("ForceAndJointImpedanceStartStop return:",error)

error = robot.GetForceAndTorqueDragState()
print("GetForceAndTorqueDragState return:",error)

time.sleep(10)

status = 0 #控制状态，0-关闭；1-开启
impedanceFlag = 0 #阻抗开启标志，0-关闭；1-开启
error = robot.ForceAndJointImpedanceStartStop(status, impedanceFlag, lamdeDain, KGain,
BGain,dragMaxTcpVel,dragMaxTcpOriVel)
print("ForceAndJointImpedanceStartStop return:",error)

error = robot.GetForceAndTorqueDragState()
print("GetForceAndTorqueDragState return:",error)

error = robot.DragTeachSwitch(0)
print("DragTeachSwitch 0  return:",error)
```

10.29. 获取力传感器拖动开关状态

在 python 版本加入: SDK-v2.0.5

原型	GetForceAndTorqueDragState()
描述	获取力传感器拖动开关状态
必选参数	NULL
默认参数	NULL
返回值	• 错误码 成功-0 失败- errcode • `返回值（调用成功返回）` `dragState`：力传感器辅助拖动控制状态，0-关闭；1-开启 • `返回值（调用成功返回）` `sixDimensionalDragState`：六维力辅助拖动控制状态，0-关闭；1-开启

10.30. 设置力传感器下负载重量

在 python 版本加入: SDK-v2.0.5

原型	SetForceSensorPayload(weight)
描述	设置力传感器下负载重量
必选参数	• `weight`：负载重量 kg
默认参数	NULL
返回值	• 错误码 成功-0 失败- errcode

10.30.1. 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')

error = robot.SetForceSensorPayload(0.8)
print("SetForceSensorPayload return:",error)

error = robot.SetForceSensorPayloadCog(0.5,0.6,12.5)
print("SetForceSensorPayLoadCog return:",error)

error = robot.GetForceSensorPayload()
print("GetForceSensorPayLoad return:",error)

error = robot.GetForceSensorPayloadCog()
print("GetForceSensorPayLoadCog return:",error)
```

10.31. 设置力传感器下负载质心

在 python 版本加入: SDK-v2.0.5

原型	SetForceSensorPayloadCog(x,y,z)
描述	设置力传感器下负载质心

必选参数	• `x`：负载质心x mm • `y`：负载质心y mm • `z`：负载质心z mm
默认参数	NULL
返回值	• 错误码 成功-0 失败- errcode

10.32. 获取力传感器下负载重量

在 python 版本加入: SDK-v2.0.5

原型	GetForceSensorPayload()
描述	获取力传感器下负载重量
必选参数	NULL
默认参数	NULL
返回值	• 错误码 成功-0 失败- errcode • 返回值（调用成功返回） `weight`：负载重量 kg

10.33. 获取力传感器下负载质心

在 python 版本加入: SDK-v2.0.5

原型	GetForceSensorPayloadCog()
描述	获取力传感器下负载质心
必选参数	NULL
默认参数	NULL
返回值	• 错误码 成功-0 失败- errcode • 返回值（调用成功返回） `x`：负载质心x mm • 返回值（调用成功返回） `y`：负载质心y mm • 返回值（调用成功返回） `z`：负载质心z mm

10.34. 力传感器自动校零

在 python 版本加入: SDK-v2.0.5

原型	ForceSensorAutoComputeLoad()
描述	力传感器自动校零
必选参数	NULL
默认参数	NULL

返回值	• 错误码 成功-0 失败- errcode • `返回值（调用成功返回） weight`：传感器质量 kg • `返回值（调用成功返回） pos=[x,y,z]`：传感器质心 mm

10.34.1. 代码示例

```python
from fairino import Robot
# 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')

error = robot.SetForceSensorPayload(0)
print("SetForceSensorPayload return:",error)

error = robot.SetForceSensorPayloadCog(0,0,0)
print("SetForceSensorPayLoadCog return:",error)

error = robot.ForceSensorAutoComputeLoad()
print("ForceSensorAutoComputeLoad return:",error)
```