5. 机器人安全设置

5.1. 设置碰撞等级

原型	`SetAnticollision (mode,level,config)`
描述	设置碰撞等级
必选参数	• `mode`:0–等级，1–百分比； • `level=[j1,j2,j3,j4,j5,j6]`:碰撞阈值； • `config`:0–不更新配置文件，1–更新配置文件
默认参数	无
返回值	错误码 成功–0 失败– errcode

5.1.1. 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')
level = [1.0,2.0,3.0,4.0,5.0,6.0]
error = robot.SetAnticollision(0,level,1)
print("设置碰撞等级错误码:",error)
level = [50.0,20.0,30.0,40.0,50.0,60.0]
error = robot.SetAnticollision(1,level,1)
print("设置碰撞等级错误码:",error)
```

5.2. 设置碰撞后策略

在 Python 版本发生变更: SDK–v2.0.8–3.7.8

`SetCollisionStrategy(strategy,safeTime,safeDistance,safeVel,safetyMargin)`

设置碰撞后策略

- `strategy`：0–报错暂停，1–继续运行，2–报错停止，3–重力矩模式，4–震荡相应模式，5–碰撞

- `safeTime`：安全停止时间[1000–2000]ms，默认为：1000
- `safeDistance`：安全停止距离[1–150]mm，默认为：100
- `safeVel`：安全停止速度[50–250]mm/s，默认为：250
- `safetyMargin[6]`：安全系数[1–10]，默认为：[10,10,10,10,10,10]

错误码 成功-0 失败- errcode

5.2.1. 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')
error = robot.SetCollisionStrategy(strategy=1)
print("设置碰撞后策略错误码:",error)
```

5.3. 设置正限位

原型	`SetLimitPositive(p_limit)`
描述	设置正限位
必选参数	• `p_limit=[j1,j2,j3,j4,j5,j6]`：六个关节位置
默认参数	无
返回值	错误码 成功-0 失败- errcode

5.3.1. 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_limit = [170.0,80.0,150.0,80.0,170.0,160.0]
error = robot.SetLimitPositive(p_limit)
print("设置正限位错误码:",error)
```

5.4. 设置负限位

原型	`SetLimitNegative(n_limit)`
描述	设置负限位
必选参数	• `n_limit=[j1,j2,j3,j4,j5,j6]`：六个关节位置
默认参数	无
返回值	错误码 成功-0 失败- errcode

5.4.1. 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')
n_limit = [-170.0,-260.0,-150.0,-260.0,-170.0,-160.0]
error = robot.SetLimitNegative(n_limit)
print("设置负限位错误码:",error)
```

5.5. 错误状态清除

原型	`ResetAllError()`
描述	错误状态清除，只能清除可复位的错误
必选参数	无
默认参数	无
返回值	错误码 成功–0 失败– errcode

5.5.1. 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')
error = robot.ResetAllError()
print("错误状态清除错误码:",error)
```

5.6. 关节摩擦力补偿开关

原型	`FrictionCompensationOnOff(state)`
描述	关节摩擦力补偿开关
必选参数	• `state`：0–关，1–开
默认参数	无
返回值	错误码 成功–0 失败– errcode

5.6.1. 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')
error = robot.FrictionCompensationOnOff(1)
print("关节摩擦力补偿开关错误码:",error)
```

5.7. 设置关节摩擦力补偿系数-正装

原型	`SetFrictionValue_level(coeff)`
描述	设置关节摩擦力补偿系数-固定安装-正装
必选参数	• `coeff=[j1,j2,j3,j4,j5,j6]`：六个关节补偿系数
默认参数	无
返回值	错误码 成功-0 失败- errcode

5.7.1. 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')
lcoeff = [0.9,0.9,0.9,0.9,0.9,0.9]
error = robot.SetFrictionValue_level(lcoeff)
print("设置关节摩擦力补偿系数-正装错误码:",error)
```

5.8. 设置关节摩擦力补偿系数-侧装

原型	`SetFrictionValue_wall(coeff)`
描述	设置关节摩擦力补偿系数-固定安装-侧装
必选参数	• `coeff=[j1,j2,j3,j4,j5,j6]`：六个关节补偿系数
默认参数	无
返回值	错误码 成功-0 失败- errcode

5.8.1. 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')
wcoeff = [0.4,0.4,0.4,0.4,0.4,0.4]
error = robot.SetFrictionValue_wall(wcoeff)
print("设置关节摩擦力补偿系数-侧装错误码:",error)
```

5.9. 设置关节摩擦力补偿系数-倒装

原型	`SetFrictionValue_ceiling(coeff)`
描述	设置关节摩擦力补偿系数–固定安装–倒装
必选参数	• `coeff=[j1,j2,j3,j4,j5,j6]`：六个关节补偿系数
默认参数	无
返回值	错误码 成功–0 失败– errcode

5.9.1. 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')
ccoeff = [0.6,0.6,0.6,0.6,0.6,0.6]
error =robot.SetFrictionValue_ceiling(ccoeff)
print("设置关节摩擦力补偿系数–倒装错误码:",error)
```

5.10. 设置关节摩擦力补偿系数-自由安装

原型	`SetFrictionValue_freedom(coeff)`
描述	设置关节摩擦力补偿系数–自由安装
必选参数	• `coeff=[j1,j2,j3,j4,j5,j6]`：六个关节补偿系数
默认参数	无
返回值	错误码 成功–0 失败– errcode

5.10.1. 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')
fcoeff = [0.5,0.5,0.5,0.5,0.5,0.5]
error =robot.SetFrictionValue_freedom(fcoeff)
print("设置关节摩擦力补偿系数–自由装错误码:",error)
```

5.11. 下载点位表数据库

latest

在 python 版本加入: SDK–v2.0.1

原型	`PointTableDownLoad(point_table_name,save_file_path)`
描述	下载点位表数据库
必选参数	• `point_table_name`：要下载的点位表名称 pointTable1.db; • `save_file_path`:下载点位表的存储路径 C://test/;
默认参数	无
返回值	错误码 成功-0 失败- errcode

5.11.1. 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')
error =
robot.PointTableDownLoad("point_table_a.db","D://Desktop/testPoint/download/")
print("PointTableDownLoad错误码:",error)
```

5.12. 上传点位表数据库

在 python 版本加入: SDK-v2.0.1

原型	`PointTableUpLoad(point_table_file_path)`
描述	上传点位表数据库
必选参数	• `point_table_file_path`：上传点位表的全路径名 C://test/pointTable1.db
默认参数	无
返回值	错误码 成功-0 失败- errcode

5.12.1. 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')
error = robot.PointTableUpLoad("D://Desktop/testPoint/point_table_a.db")
print("PointTableUpLoad错误码:",error)
```

5.13. 点位表切换

在 python 版本加入: SDK-v2.0.1

`ableSwitch(point_table_name)`	
切换	
`int_table_name`：要切换的点位表名称pointTable1.db,当点位表为空，即”时，表示将lua程序更新为未应	
成功–0 失败– errcode	

5.13.1. 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')
error = robot.PointTableSwitch("point_table_a.db")
print("PointTableSwitch:",error)
```

5.14. 点位表更新lua文件

在 python 版本加入: SDK–v2.0.1

`ableUpdateLua(point_table_name, lua_file_name)`	
更新lua文件	
`int_table_name`：要切换的点位表名称 pointTable1.db,当点位表为空，即”时，表示将lua程序更新为未应 `_file_name`：要更新的lua文件名称 testPointTable.lua	
成功–0 失败– errcode	

5.14.1. 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')
error = robot.PointTableUpdateLua("point_table_a.db","testpoint.lua")
print("PointTableUpdateLua:",error)
```

5.15. 设置机器人碰撞检测方法

在 python 版本加入: SDK–v2.0.5

latest

原型	`SetCollisionDetectionMethod(method)`

描述	设置机器人碰撞检测方法
必选参数	• `method`：碰撞检测方法：0–电流模式；1–双编码器；2–电流和双编码器同时开启
默认参数	无
返回值	• 错误码 成功–0 失败– errcode

5.16. 设置静态下碰撞检测开始关闭

在 python 版本加入: SDK–v2.0.5

原型	`SetStaticCollisionOnOff(status)`
描述	设置静态下碰撞检测开始关闭
必选参数	• `status`： 0–关闭；1–开启
默认参数	无
返回值	• 错误码 成功–0 失败– errcode

5.17. 设置碰撞检测开始关闭

在 python 版本加入: SDK–v2.0.5

原型	`SetPowerLimit(status, power)`
描述	设置静态下碰撞检测开始关闭
必选参数	• `status`： 0–关闭；1–开启
默认参数	无
返回值	• 错误码 成功–0 失败– errcode

5.17.1. 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.SetPowerLimit(0,2)
print("SetPowerLimit return:",error)

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

error,joint_torque = robot.GetJointTorques()
print("GetJointTorques return",joint_torque)
joint_torque =
[joint_torque[0],joint_torque[1],joint_torque[2],joint_torque[3],joint_torque[4],joint_t
error_joint = 0
count =100
error = robot.ServoJTStart()    #servoJT开始
print("ServoJTStart return",error)
while(count):
    if error!=0:
        error_joint =error
    joint_torque[0] = joint_torque[0] + 10  #每次1轴增加0.1NM，运动100次
    error = robot.ServoJT(joint_torque, 0.001)  # 关节空间伺服模式运动
    count = count - 1
    time.sleep(0.001)
print("ServoJTStart return",error_joint)
error = robot.ServoJTEnd()  #伺服运动结束
time.sleep(1)
print("ServoJTEnd return",error)
```

5.18. 奇异位姿保护开启

在 python 版本加入: SDK-v2.0.5

原型	`SingularAvoidStart(protectMode, minShoulderPos=100, minElbowPos=50, minWristPos=10)`
描述	开启奇异位姿保护
必选参数	• `protectMode`：奇异位姿保护保护模式：0-关节模式；1-笛卡尔模式
默认参数	• `minShoulderPos`：肩奇异调整范围(mm), 默认100.0 • `minElbowPos`：肘奇异调整范围(mm), 默认50.0 • `minWristPos`：腕奇异调整范围(°), 默认10.0
返回值	• 错误码 成功-0 失败- errcode

5.19. 奇异位姿保护关闭

latest

在 python 版本加入: SDK-v2.0.5

原型	`SingularAvoidEnd()`
描述	关闭奇异位姿保护
必选参数	无
默认参数	无
返回值	• 错误码 成功–0 失败– errcode

5.19.1. 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')

startdescPose = [-352.437, -88.350, 226.471, 177.222, 4.924, 86.631]
startjointPos = [-3.463, -84.308, 105.579, -108.475, -85.087, -0.334]

middescPose = [-518.339, -23.706, 207.899, -178.420, 0.171, 71.697]
midjointPos = [-8.587, -51.805, 64.914, -104.695, -90.099, 9.718]

enddescPose = [-273.934, 323.003, 227.224, 176.398, 2.783, 66.064]
endjointPos = [-63.460, -71.228, 88.068, -102.291, -90.149, -39.605]

robot.MoveL(desc_pos=startdescPose, tool=0, user=0,vel=50)
error = robot.SingularAvoidStart(1,100,50,10)
print("SingularAvoidStart return ", error)

robot.MoveC(desc_pos_p=middescPose,tool_p=0,user_p=0,desc_pos_t=enddescPose,tool_t=0,use
error = robot.SingularAvoidEnd()
print("SingularAvoidEnd return ", error)
```

5.20. 自定义碰撞检测阈值功能开始，设置关节端和TCP端的碰撞检测阈值

在 python 版本加入: SDK–v2.1.0

原型	`CustomCollisionDetectionStart(flag, jointDetectionThreshould, tcpDetectionThreshould, block)`
描述	自定义碰撞检测阈值功能开始，设置关节端和TCP端的碰撞检测阈值
必选参数	• `flag`： 1–仅关节检测开启；2–仅TCP检测开启；3–关节和TCP检测同时开启 • `jointDetectionThreshould`： 关节碰撞检测阈值 j1–j6 • `tcpDetectionThreshould`： TCP碰撞检测阈值，xyzabc • `block`： 0–非阻塞；1–阻塞
默认参数	无
返回值	• 错误码 成功–0 失败– errcode

5.21. 自定义碰撞检测阈值功能关闭

在 python 版本加入: SDK-v2.1.0

原型	`CustomCollisionDetectionEnd()`
描述	自定义碰撞检测阈值功能关闭
必选参数	无
默认参数	无
返回值	• 错误码 成功-0 失败- errcode

5.21.1. 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')
safety = [5, 5, 5, 5, 5, 5]
robot.SetCollisionStrategy(3, 1000, 150, 250, safety)
jAointDetectionThreshould = [0.3, 0.3, 0.3, 0.3, 0.3, 0.3]
tcpDetectionThreshould = [80, 80, 80, 80, 80, 80]
rtn = robot.CustomCollisionDetectionStart(3, jAointDetectionThreshould, tcpDetectionThreshould, 0)
print("CustomCollisionDetectionStart rtn is ", rtn)
p1Desc = [228.879, -503.594, 453.984, -175.580, 8.293, 171.267]
p1Joint = [102.700, -85.333, 90.518, -102.365, -83.932, 22.134]

p2Desc = [-333.302, -435.580, 449.866, -174.997, 2.017, 109.815]
p2Joint = [41.862, -85.333, 90.526, -100.587, -90.014, 22.135]

exaxisPos = [0.0, 0.0, 0.0, 0.0]
offdese = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
while True:
    robot.MoveL(desc_pos=p1Desc, tool=0, user=0, vel=100, acc=100, ovl=100)
    robot.MoveL(desc_pos=p2Desc, tool=0, user=0, vel=100, acc=100, ovl=100)
rtn = robot.CustomCollisionDetectionEnd()
print("CustomCollisionDetectionEnd rtn is ", rtn)
```