4. 机器人常用设置

4.1. 设置全局速度

原型	`SetSpeed(vel)`
描述	设置全局速度
必选参数	• `vel` :速度百分比，范围[0~100]
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.1.1. 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')
error = robot.SetSpeed(20)
print("设置全局速度错误码:",error)
```

4.2. 设置系统变量值

原型	`SetSysVarValue(id,value)`
描述	设置系统变量
必选参数	• `id` ：变量编号，范围[1~20]; • `value` ：变量值
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.2.1. 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')
for i in range(1,21):
    error = robot.SetSysVarValue(i,10)
robot.WaitMs(1000)
for i in range(1,21):
    sys_var = robot.GetSysVarValue(i)
    print("系统变量编号:",i,"值",sys_var)
```

4.3. 设置工具参考点-六点法

原型	`SetToolPoint(point_num)`
描述	设置工具参考点–六点法
必选参数	• `point_num`：点编号,范围[1~6]
默认参数	无
返回值	错误码 成功–0 失败– errcode

4.3.1. 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')
t_coord = [1.0,2.0,3.0,4.0,5.0,6.0]
for i in range(1,7):
    robot.DragTeachSwitch(1)#切入拖动示教模式
    time.sleep(5)
    error = robot.SetToolPoint(i) #实际应当控制机器人按照要求移动到合适位置后再发送指令
    print("六点法设置工具坐标系，记录点",i,"错误码",error)
    robot.DragTeachSwitch(0)
    time.sleep(1)
error = robot.ComputeTool()
print("六点法设置工具坐标系错误码",error)
```

4.4. 计算工具坐标系-六点法

原型	`ComputeTool()`
描述	计算工具坐标系–六点法（设置完六个工具参考点后再进行计算）
必选参数	无
默认参数	无

返回值	• 错误码 成功–0 失败– errcode • `tcp_pose=[x,y,z,rx,ry,rz]`：工具坐标系

4.5. 设置工具参考点-四点法

原型	`SetTcp4RefPoint(point_num)`
描述	设置工具参考点–四点法
必选参数	`point_num`：点编号,范围[1~4]
默认参数	无
返回值	• 错误码 成功–0 失败– errcode • `tcp_pose=[x,y,z,rx,ry,rz]`：工具坐标系

4.5.1. 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')
t_coord = [1.0,2.0,3.0,4.0,5.0,6.0]
for i in range(1,5):
    robot.DragTeachSwitch(1)#切入拖动示教模式
    time.sleep(5)
    error = robot.SetTcp4RefPoint(i) #应当控制机器人按照要求移动到合适位置后再发送指令
    print("四点法设置工具坐标系，记录点",i,"错误码",error)
    robot.DragTeachSwitch(0)
    time.sleep(1)
error,t_coord= robot.ComputeTcp4()
print("四点法设置工具坐标系错误码",error,"工具TCP",t_coord)
```

4.6. 计算工具坐标系-四点法

原型	`ComputeTcp4()`
描述	计算工具坐标系–四点法（设置完四个工具参考点后再进行计算）
必选参数	无
默认参数	无
返回值	• 错误码 成功–0 失败– errcode • `tcp_pose=[x,y,z,rx,ry,rz]`：工具坐标系

4.7. 设置工具坐标系

原型	`SetToolCoord(id,t_coord,type,install,toolID,loadNum)`
描述	设置工具坐标系
必选参数	• `id`:坐标系编号，范围[1~15]； • `t_coord`:工具中心点相对末端法兰中心位姿，单位[mm][°]； • `type`:0–工具坐标系，1–传感器坐标系； • `install`:安装位置，0–机器人末端，1–机器人外部 • `toolID`:工具ID • `loadNum`:负载编号
默认参数	无
返回值	错误码 成功–0 失败– errcode

4.7.1. 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')
t_coord = [1.0,2.0,3.0,4.0,5.0,6.0]
error = robot.SetToolCoord(10,t_coord,0,0,0,0)
print("设置工具坐标系错误码",error)
```

4.8. 设置工具坐标系列表

原型	`SetToolList(id,t_coord ,type, install, loadNum)`
描述	设置工具坐标系列表
必选参数	• `id`:坐标系编号，范围[1~15]； • `t_coord`:[x,y,z,rx,ry,rz] 工具中心点相对末端法兰中心位姿，单位[mm][°]； • `type`:0–工具坐标系，1–传感器坐标系； • `install`:安装位置，0–机器人末端，1–机器人外部 • `loadNum`:负载编号
默认参数	无
返回值	错误码 成功–0 失败– errcode

4.8.1. 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')
t_coord = [1.0,2.0,3.0,4.0,5.0,6.0]
error = robot.SetToolList(10,t_coord,0,0,0)
print("设置工具坐标系列表错误码",error)
```

4.9. 设置外部工具参考点-三点法

原型	`SetExTCPPoint(point_num)`
描述	设置外部工具参考点–三点法
必选参数	• `point_num`：点编号,范围[1~3]
默认参数	无
返回值	错误码 成功–0 失败– errcode

4.9.1. 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')
etcp = [1.0,2.0,3.0,4.0,5.0,6.0]
etool = [21.0,22.0,23.0,24.0,25.0,26.0]
for i in range(1,4):
    error = robot.SetExTCPPoint(i) #应当控制机器人按照要求移动到合适位置后再发送指令
    print("三点法设置外部工具坐标系，记录点",i,"错误码",error)
    time.sleep(1)
error,etcp = robot.ComputeExTCF()
print("三点法设置外部工具坐标系错误码",error,"外部工具TCP",etcp)
error = robot.SetExToolCoord(10,etcp,etool)
print("设置外部工具坐标系错误码",error)
error = robot.SetExToolList(10,etcp,etool)
print("设置外部工具坐标系列表错误码",error)
```

4.10. 计算外部工具坐标系-三点法

原型	`ComputeExTCF (point_num)`
描述	计算外部工具坐标系–三点法（设置完三个参考点后再进行计算）
必选参数	`point_num`：点编号,范围[1~3]
默认参数	无

返回值	• 错误码 成功-0 失败- errcode • `etcp=[x,y,z,rx,ry,rz]`：外部工具坐标系

4.11. 设置外部工具坐标系

原型	`SetExToolCoord(id,etcp,etool)`
描述	设置外部工具坐标系
必选参数	• `id`:坐标系编号，范围[0~14]； • `etcp`:外部工具坐标系，单位[mm][°]； • `etool`:末端工具坐标系，单位[mm][°]；
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.11.1. 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')
etcp = [1.0,2.0,3.0,4.0,5.0,6.0]
etool = [21.0,22.0,23.0,24.0,25.0,26.0]
error = robot.SetExToolCoord(10,etcp,etool)
print("设置外部工具坐标系错误码",error)
```

4.12. 设置外部工具坐标系列表

原型	`SetExToolList(id,etcp ,etool)`
描述	设置外部工具坐标系列表
必选参数	• `id`:坐标系编号，范围[0~14]； • `etcp`:外部工具坐标系，单位[mm][°]； • `etool`:末端工具坐标系，单位[mm][°]；
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.12.1. 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')
etcp = [1.0,2.0,3.0,4.0,5.0,6.0]
etool = [21.0,22.0,23.0,24.0,25.0,26.0]
error = robot.SetExToolList(10,etcp,etool)
print("设置外部工具坐标系列表错误码",error)
```

4.13. 设置工件参考点-三点法

原型	`SetWObjCoordPoint(point_num)`
描述	设置工件参考点–三点法
必选参数	`point_num` :点编号,范围[1~3]
默认参数	无
返回值	错误码 成功–0 失败– errcode

4.13.1. 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')
w_coord = [11.0,12.0,13.0,14.0,15.0,16.0]
robot.SetToolList(0,[0,0,0,0,0,0],0,0)#设置参考点前应当将工具和工件号坐标系切换至0
robot.SetWObjList(0,[0,0,0,0,0,0])
for i in range(1,4):
    error = robot.SetWObjCoordPoint(i) #实际应当控制机器人按照要求移动到合适位置后再发送指令
    print("三点法设置工件坐标系，记录点",i,"错误码",error)
    time.sleep(1)
error, w_coord = robot.ComputeWObjCoord(0,0)
print("三点法计算工件坐标系错误码",error,"工件坐标系", w_coord)
```

4.14. 计算工件坐标系-三点法

原型	`ComputeWObjCoord(method, refFrame)`
描述	计算工件坐标系–三点法（三个参考点设置完成后再进行计算）；
必选参数	• `method`：计算方式0：原点–x轴–z轴，1：原点–x轴–xy平面 • `refFrame`：参考坐标系
默认参数	无

返回值	• 错误码 成功-0 失败- errcode • `wobj_pose=[x,y,z,rx,ry,rz]`：工件坐标系

4.15. 设置工件坐标系

原型	`SetWObjCoord(id, coord, refFrame)`
描述	设置工件坐标系
必选参数	• `id`:坐标系编号，范围[0~14]； • `coord`:工件坐标系相对于末端法兰中心位姿，单位 [mm][°] • `refFrame`:参考坐标系
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.15.1. 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')
w_coord = [11.0,12.0,13.0,14.0,15.0,16.0]
error = robot.SetWObjCoord(id=11,coord=w_coord,refFrame=0)
print("设置工件坐标系错误码",error)
```

4.16. 设置工件坐标系列表

原型	`SetWObjList(id, coord, refFrame)`
描述	设置工件坐标系列表
必选参数	• `id`:坐标系编号，范围[0~14]； • `coord`:工件坐标系相对于末端法兰中心位姿，单位 [mm][°] • `refFrame`:参考坐标系
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.16.1. 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')
w_coord = [11.0,12.0,13.0,14.0,15.0,16.0]
error = robot.SetWObjList(id=11,coord=w_coord,refFrame=0)
print("设置工件坐标系列表错误码",error)
```

4.17. 设置末端负载重量

在 Python 版本发生变更: SDK-v2.0.8-3.7.8

原型	`SetLoadWeight(loadNum, weight)`
描述	设置末端负载重量,错误负载重量设置可能会导致拖动模式下机器人失控
必选参数	• `loadNum`:负载编号 • `weight`:单位[kg]
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.17.1. 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.SetLoadWeight(0,0)#！！！负载重量设置应于实际相符(错误负载重量设置可能会导致拖动模式下机器人失控)
```

4.18. 设置机器人安装方式-固定安装

原型	`SetRobotInstallPos(method)`
描述	设置机器人安装方式-固定安装,错误安装方式设置会导致拖动模式下机器人失控
必选参数	• `method`:0-平装，1-侧装，2-挂装
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.18.1. 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.SetRobotInstallPos(0) #！！！安装方式设置应与实际一致 0-正装，1-侧装，2-倒装（错误安装方式设置会导致拖动模式下机器人失控）
print("设置机器人安装方式错误码",error)
```

4.19. 设置机器人安装角度-自由安装

原型	`SetRobotInstallAngle(yangle,zangle)`
描述	设置机器人安装角度-自由安装,错误安装角度设置会导致拖动模式下机器人失控
必选参数	• `yangle`：倾斜角 • `zangle`：旋转角
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.19.1. 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.SetRobotInstallAngle(0.0,0.0) #！！！安装角度设置应与实际一致（错误安装角度设置会导致拖动模式下机器人失控）
print("设置机器人安装角度错误码",error)
```

4.20. 设置末端负载质心坐标

原型	`SetLoadCoord(x,y,z)`
描述	设置末端负载质心坐标,错误负载质心设置可能会导致拖动模式下机器人失控
必选参数	• `x`: 质心坐标，单位[mm] • `y`: 质心坐标，单位[mm] • `z`: 质心坐标，单位[mm]
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.20.1. 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.SetLoadCoord(3.0,4.0,5.0) #！！！负载质心设置应于实际相符(错误负载质心设置可能会导致拖动模式下机器人失控)
print("设置负载质心错误码",error)
```

4.21. 等待指定时间

原型	`WaitMs(t_ms)`
描述	等待指定时间
必选参数	• `t_ms`:单位[ms]
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.21.1. 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.WaitMs(1000)
print("等待指定时间错误码",error)
```

4.22. 设置机器人加速度

在 python 版本加入: SDK-v2.0.4

原型	`SetOaccScale(acc)`
描述	设置机器人加速度
必选参数	• `acc`:机器人加速度百分比
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.22.1. 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')
robot.SetOaccScale (20)
```

4.23. 设置机器指定姿态速度开启

在 python 版本加入: SDK-v2.0.5

原型	`AngularSpeedStart(ratio)`
描述	指定姿态速度开启
必选参数	• `ratio`:姿态速度百分比[0-300]
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.24. 指定姿态速度关闭

在 python 版本加入: SDK-v2.0.5

原型	`AngularSpeedEnd()`
描述	指定姿态速度关闭
必选参数	无
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.25. 工具坐标系转换开始

在 Python 版本加入: SDK-v2.0.8-3.7.8

原型	`ToolTrsfStart(toolNum)`
描述	工具坐标系转换开始
必选参数	• `toolNum`:工具坐标系编号[0-14]
默认参数	无
返回值	错误码 成功-0 失败- errcode

4.26. 工具坐标系转换结束

在 Python 版本加入: SDK–v2.0.8–3.7.8

原型	`ToolTrsfEnd()`
描述	工具坐标系转换结束
必选参数	无
默认参数	无
返回值	错误码 成功–0 失败– errcode

4.26.1. 代码示例

在 Python 版本加入: SDK–v2.0.8–3.7.8

```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')

startjointPos = [52.850, -84.327, 102.163, -112.843, -84.131, 0.063]
startdescPose = [-226.699, -501.969, 264.638, -174.973, 5.852, 143.301]
endjointPos = [52.850, -77.596, 111.785, -129.196, -84.131, 0.062]
enddescPose = [-226.702, -501.973, 155.833, -174.973, 5.852, 143.301]

robot.ToolTrsfStart(1)
rtn = robot.MoveJ(startjointPos, 0, 0, startdescPose)
print("rtn is ", rtn)
rtn = robot.MoveJ(endjointPos, 0, 0, enddescPose)
print("rtn is ", rtn)
robot.ToolTrsfEnd()
```

4.27. 根据点位信息计算工具坐标系

在 Python 版本加入: SDK–v2.0.8–3.7.8

原型	`ComputeToolCoordWithPoints(method, pos)`
描述	根据点位信息计算工具坐标系
必选参数	• `method`：计算方法；0–四点法；1–六点法 • `pos`：关节位置组，四点法时数组长度为4个，六点法时数组长度为6个
默认参数	无
返回值	• 错误码 成功–0 失败– errcode • `tcp_offset=[x,y,z,rx,ry,rz]`：根据点位信息计算得到的工具坐标系，单位 [mm][°]

4.27.1. 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')

p1Desc = [-394.073, -276.405, 399.451, -133.692, 7.657, -139.047]
p1Joint = [15.234, -88.178, 96.583, -68.314, -52.303, -122.926]

p2Desc = [-187.141, -444.908, 432.425, 148.662, 15.483, -90.637]
p2Joint = [61.796, -91.959, 101.693, -102.417, -124.511, -122.767]

p3Desc = [-368.695, -485.023, 426.640, -162.588, 31.433, -97.036]
p3Joint = [43.896, -64.590, 60.087, -50.269, -94.663, -122.652]

p4Desc = [-291.069, -376.976, 467.560, -179.272, -2.326, -107.757]
p4Joint = [39.559, -94.731, 96.307, -93.141, -88.131, -122.673]

p5Desc = [-284.140, -488.041, 478.579, 179.785, -1.396, -98.030]
p5Joint = [49.283, -82.423, 81.993, -90.861, -89.427, -122.678]

p6Desc = [-296.307, -385.991, 484.492, -178.637, -0.057, -107.059]
p6Joint = [40.141, -92.742, 91.410, -87.978, -88.824, -122.808]

exaxisPos = [0, 0, 0, 0]
offdese = [0, 0, 0, 0, 0, 0]

posJ = [p1Joint, p2Joint, p3Joint, p4Joint, p5Joint, p6Joint]
rtn, coordRtn = robot.ComputeToolCoordWithPoints(1, posJ)
print("ComputeToolCoordWithPoints ", rtn, "coord is ", coordRtn[0], coordRtn[1], coordRtn[2], coordRtn[3], coordRtn[4], coordRtn[5])

robot.MoveJ(p1Joint, 0, 0, p1Desc)
robot.SetToolPoint(1)
robot.MoveJ(p2Joint, 0, 0, p2Desc)
robot.SetToolPoint(2)
robot.MoveJ(p3Joint, 0, 0, p3Desc)
robot.SetToolPoint(3)
robot.MoveJ(p4Joint, 0, 0, p4Desc)
robot.SetToolPoint(4)
robot.MoveJ(p5Joint, 0, 0, p5Desc)
robot.SetToolPoint(5)
robot.MoveJ(p6Joint, 0, 0, p6Desc)
robot.SetToolPoint(6)
rtn, coordRtn = robot.ComputeTool()
print("ComputeTool ", rtn, "coord is ", coordRtn[0], coordRtn[1], coordRtn[2], coordRtn[3], coordRtn[4], coordRtn[5])
```

4.28. 根据点位信息计算工件坐标系

在 Python 版本加入: SDK-v2.0.8-3.7.8

原型	`ComputeWObjCoordWithPoints(method, pos, refFrame)`
描述	根据点位信息计算工件坐标系

必选参数	• `method`：计算方法；0：原点–x轴–z轴 1：原点–x轴–xy平面 • `pos`：三个TCP位置组 • `refFrame`：参考坐标系
默认参数	无
返回值	• 错误码 成功–0 失败– errcode • `wobj_offset=[x,y,z,rx,ry,rz]`：根据点位信息计算得到的工件坐标系，单位 [mm][°]

4.28.1. 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')

p1Desc = [-275.046, -293.122, 28.747, 174.533, -1.301, -112.101]
p1Joint = [35.207, -95.350, 133.703, -132.403, -93.897, -122.768]

p2Desc = [-280.339, -396.053, 29.762, 174.621, -3.448, -102.901]
p2Joint = [44.304, -85.020, 123.889, -134.679, -92.658, -122.768]

p3Desc = [-270.597, -290.603, 83.034, 179.314, 0.808, -114.171]
p3Joint = [32.975, -99.175, 125.966, -116.484, -91.014, -122.857]

exaxisPos = [0, 0, 0, 0]
offdese = [0, 0, 0, 0, 0, 0]

posTCP = [p1Desc, p2Desc, p3Desc]
rtn, coordRtn = robot.ComputeWObjCoordWithPoints(1, posTCP, 0)
print("ComputeWObjCoordWithPoints ", rtn, "coord is ", coordRtn[0], coordRtn[1], coordRtn[2], coordRtn[3], coordRtn[4], coordRtn[5])

robot.MoveJ(p1Joint, 1, 0, p1Desc)
robot.SetWObjCoordPoint(1)
robot.MoveJ(p2Joint, 1, 0, p2Desc)
robot.SetWObjCoordPoint(2)
robot.MoveJ(p3Joint, 1, 0, p3Desc)
robot.SetWObjCoordPoint(3)
rtn, coordRtn = robot.ComputeWObjCoord(1, 0)
print("ComputeTool ", rtn, "coord is ", coordRtn[0], coordRtn[1], coordRtn[2], coordRtn[3], coordRtn[4], coordRtn[5])
```