

机器人C#SDK接口更新说明(V3.8.0)

**修 订 记 录**

|  |  |  |
| --- | --- | --- |
| 日期 | Web版本 | 描述 |
| 2025/02/08 | V3.7.8 | 创建文件 |
| 2025/02/12 | V3.8.0 | 1. EndForceDragControl()接口增加奇异点规避参数 2. ArcWeldTraceControl()接口增加偏置参数 3. 增加WeaveChangeStart()接口 4. 增加WeaveChangeEnd()接口 5. 增加LoadTrajectoryLA()接口 6. 增加MoveTrajectoryLA()接口 7. 增加CustomCollisionDetectionStart()接口 8. 增加CustomCollisionDetectionEnd()接口 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**说明：**

**①本手册适用于WebApp版本3.8.0的协作机器人，手册的内容如有变更，恕不另行通知。查看其它版本请登录法奥文档：**

<https://fr-documentation.readthedocs.io/zh_CN/latest/index.html>

目 录

[1 参数更改接口 1](#_Toc5605)

[1.1 力传感器辅助拖动 1](#_Toc22584)

[1.1.1 接口描述 1](#_Toc11962)

[1.1.2 代码示例 1](#_Toc25778)

[1.2 电弧跟踪控制 2](#_Toc8325)

[1.2.1 接口描述 2](#_Toc30031)

[1.2.2 代码示例 3](#_Toc14807)

[1.3 C#SDK接口说明文档位置调整 4](#_Toc20603)

[2 新增接口 4](#_Toc6407)

[2.1 摆动渐变开始、摆动渐变结束 4](#_Toc11876)

[2.1.1 接口描述 4](#_Toc9211)

[2.1.2 代码示例 4](#_Toc23070)

[2.2 轨迹预处理(轨迹前瞻)、轨迹复现(轨迹前瞻) 5](#_Toc2861)

[2.3 接口描述 5](#_Toc29964)

[2.3.1 代码示例 6](#_Toc6655)

[2.4 自定义碰撞检测阈值功能开始、结束 6](#_Toc13926)

[2.4.1 接口描述 6](#_Toc26271)

[2.4.2 代码示例 7](#_Toc4116)

# 参数更改接口

## 力传感器辅助拖动

### 接口描述

章节：11.25 力传感器辅助拖动

|  |
| --- |
| 1. /\*\* 2. \* @brief 力传感器辅助拖动 3. \* @param [in] status 控制状态，0-关闭；1-开启 4. \* @param [in] asaptiveFlag 自适应开启标志，0-关闭；1-开启 5. \* @param [in] interfereDragFlag 干涉区拖动标志，0-关闭；1-开启 6. \* @param [in] ingularityConstraintsFlag 奇异点策略，0-规避；1-穿越 7. \* @param [in] M 惯性系数 8. \* @param [in] B 阻尼系数 9. \* @param [in] K 刚度系数 10. \* @param [in] F 拖动六维力阈值 11. \* @param [in] Fmax 最大拖动力限制 Nm 12. \* @param [in] Vmax 最大关节速度限制 °/s 13. \* @return 错误码 14. \*/ 15. public int EndForceDragControl(int status, int asaptiveFlag, int interfereDragFlag, int ingularityConstraintsFlag, double[] M, double[] B, double[] K, double[] F, double Fmax, double Vmax) |

### 代码示例

|  |
| --- |
| 1. private void btnStopDrag\_Click(object sender, EventArgs e) 2. { 3. double[] M = new double[6] { 15.0, 15.0, 15.0, 0.5, 0.5, 0.1 }; 4. double[] B = new double[6] { 150, 150, 150, 5.0, 5.0, 1.0 }; 5. double[] K = new double[6] { 0, 0, 0, 0, 0, 0 }; 6. double[] F = new double[6] { 10, 10, 10, 1, 1, 1 }; 7. robot.EndForceDragControl(0, 0, 0,0, M, B, K, F, 50, 100); 8. } |

## 电弧跟踪控制

### 接口描述

章节：12.23 电弧跟踪控制

|  |
| --- |
| 1. /\*\* 2. \* @brief 电弧跟踪控制 3. \* @param [in] flag 开关，0-关；1-开 4. \* @param [in] dalayTime 滞后时间，单位ms 5. \* @param [in] isLeftRight 左右偏差补偿 6. \* @param [in] klr 左右调节系数(灵敏度) 7. \* @param [in] tStartLr 左右开始补偿时间cyc 8. \* @param [in] stepMaxLr 左右每次最大补偿量 mm 9. \* @param [in] sumMaxLr 左右总计最大补偿量 mm 10. \* @param [in] isUpLow 上下偏差补偿 11. \* @param [in] kud 上下调节系数(灵敏度) 12. \* @param [in] tStartUd 上下开始补偿时间cyc 13. \* @param [in] stepMaxUd 上下每次最大补偿量 mm 14. \* @param [in] sumMaxUd 上下总计最大补偿量 15. \* @param [in] axisSelect 上下坐标系选择，0-摆动；1-工具；2-基座 16. \* @param [in] referenceType 上下基准电流设定方式，0-反馈；1-常数 17. \* @param [in] referSampleStartUd 上下基准电流采样开始计数(反馈)，cyc 18. \* @param [in] referSampleCountUd 上下基准电流采样循环计数(反馈)，cyc 19. \* @param [in] referenceCurrent 上下基准电流mA 20. \* @param [in] offsetType 偏置跟踪类型，0-不偏置；1-采样；2-百分比 /version 3.7.9 21. \* @param [in] offsetParameter 偏置参数；采样(偏置采样开始时间，默认采一周期)；百分比(偏置百分比(-100 ~ 100)) /version 3.7.9 22. \* @return 错误码 23. \*/ 24. int ArcWeldTraceControl(int flag, double delaytime, int isLeftRight, double klr, double tStartLr, double stepMaxLr, double sumMaxLr, int isUpLow, double kud, double tStartUd, double stepMaxUd, double sumMaxUd, int axisSelect, int referenceType, double referSampleStartUd, double referSampleCountUd, double referenceCurrent, int offsetType, int offsetParameter) |

### 代码示例

|  |
| --- |
| 1. private void btnweld\_Click(object sender, EventArgs e) 2. { 3. //电弧跟踪 4. DescPose p1Desc = new DescPose(-72.912, -587.664, 31.849, 43.283, -6.731, 15.068); 5. JointPos p1Joint = new JointPos(74.620, -80.903, 94.608, -109.882, -90.436, -13.432); 6. DescPose p2Desc = new DescPose(-104.915, -483.712, -25.231, 42.228, -6.572, 18.433); 7. JointPos p2Joint = new JointPos(66.431, -92.875, 116.362, -120.516, -88.627, -24.731); 8. DescPose p3Desc = new DescPose(-242.834, -498.697, -23.681, 46.576, -5.286, 8.318); 9. JointPos p3Joint = new JointPos(57.153, -82.046, 104.060, -116.659, -92.478, -24.735); 10. ExaxisPos exaxisPos = new ExaxisPos(0.0, 0.0, 0.0, 0.0); 11. DescPose offdese = new DescPose(0.0, 0.0, 0.0, 0.0, 0.0, 0.0); 12. robot.WeldingSetVoltage(1, 19, 0, 0); 13. robot.WeldingSetCurrent(1, 190, 0, 0); 14. robot.MoveJ(p1Joint, p1Desc, 1, 1, 100, 100, 100, exaxisPos, -1, 0, offdese); 15. robot.MoveL(p2Joint, p2Desc, 1, 1, 100, 100, 50, -1, exaxisPos, 0, 0, offdese); 16. robot.ARCStart(1, 0, 10000); 17. robot.ArcWeldTraceControl(1, 0, 1, 0.06, 5, 5, 60, 1, 0.06, 5, 5, 60, 0, 0, 4, 1, 10, 2, 2); 18. robot.WeaveStart(0); 19. robot.MoveL(p3Joint, p3Desc, 1, 1, 100, 100, 1, -1, exaxisPos, 0, 0, offdese); 20. robot.WeaveEnd(0); 21. robot.ArcWeldTraceControl(0, 0, 1, 0.06, 5, 5, 60, 1, 0.06, 5, 5, 60, 0, 0, 4, 1, 10, 2, 2); 22. robot.ARCEnd(1, 0, 10000); 23. } |

## C#SDK接口说明文档位置调整

# 新增接口

## 摆动渐变开始、摆动渐变结束

### 接口描述

章节：12、机器人焊接 12.46 12.47

|  |
| --- |
| 1. /\*\* 2. \* @brief 摆动渐变开始 3. \* @param [in] weaveNum 摆动编号 4. \* @return 错误码 5. \* @version 6. \*/ 7. public int WeaveChangeStart(int weaveNum) 8. /\*\* 9. \* @brief 摆动渐变结束 10. \* @return 错误码 11. \* @version 3.7.9 12. \*/ 13. public int WeaveChangeEnd() |

### 代码示例

|  |
| --- |
| 1. private void btnweld\_Click(object sender, EventArgs e) 2. { 3. //摆动渐变 4. DescPose p1Desc = new DescPose(-72.912, -587.664, 31.849, 43.283, -6.731, 15.068); 5. JointPos p1Joint = new JointPos(74.620, -80.903, 94.608, -109.882, -90.436, -13.432); 6. DescPose p2Desc = new DescPose(-104.915, -483.712, -25.231, 42.228, -6.572, 18.433); 7. JointPos p2Joint = new JointPos(66.431, -92.875, 116.362, -120.516, -88.627, -24.731); 8. DescPose p3Desc = new DescPose(-240.651, -483.840, -7.161, 46.577, -5.286, 8.318); 9. JointPos p3Joint = new JointPos(56.457, -84.796, 104.618, -114.497, -92.422, -25.430); 10. ExaxisPos exaxisPos = new ExaxisPos(0.0, 0.0, 0.0, 0.0); 11. DescPose offdese = new DescPose(0.0, 0.0, 0.0, 0.0, 0.0, 0.0); 12. robot.WeldingSetVoltage(1, 19, 0, 0); 13. robot.WeldingSetCurrent(1, 190, 0, 0); 14. robot.MoveJ(p1Joint, p1Desc, 1, 1, 100, 100, 100, exaxisPos, -1, 0, offdese); 15. robot.MoveL(p2Joint, p2Desc, 1, 1, 100, 100, 50, -1, exaxisPos, 0, 0, offdese); 16. robot.ARCStart(1, 0, 10000); 17. robot.ArcWeldTraceControl(1, 0, 1, 0.06, 5, 5, 60, 1, 0.06, 5, 5, 80, 0, 0, 4, 1, 10, 0, 0); 18. robot.WeaveStart(0); 19. robot.WeaveChangeStart(1); 20. robot.MoveL(p3Joint, p3Desc, 1, 1, 100, 100, 1, -1, exaxisPos, 0, 0, offdese); 21. robot.WeaveChangeEnd(); 22. robot.WeaveEnd(0); 23. robot.ArcWeldTraceControl(0, 0, 1, 0.06, 5, 5, 60, 1, 0.06, 5, 5, 80, 0, 0, 4, 1, 10, 0, 0); 24. robot.ARCEnd(1, 0, 10000); 25. } |

## 轨迹预处理(轨迹前瞻)、轨迹复现(轨迹前瞻)

## 接口描述

章节：8、机器人轨迹复现

|  |
| --- |
| 1. /\*\* 2. \* @brief 轨迹预处理(轨迹前瞻) 3. \* @param [in] name 轨迹文件名 4. \* @param [in] 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 5. \* @param [in] errorLim 误差限制，使用直线拟合生效 6. \* @param [in] type 平滑方式，0-贝塞尔平滑 7. \* @param [in] precision 平滑精度，使用贝塞尔平滑时生效 8. \* @param [in] vamx 设定的最大速度，mm/s 9. \* @param [in] amax 设定的最大加速度，mm/s2 10. \* @param [in] jmax 设定的最大加加速度，mm/s3 11. \* @return 错误码 12. \*/ 13. int LoadTrajectoryLA(string name, int mode, double errorLim, int type, double precision, double vamx, double amax, double jmax) 14. /\*\* 15. \* @brief 轨迹复现(轨迹前瞻) 16. \* @return 错误码 17. \*/ 18. int MoveTrajectoryLA() |

### 代码示例

|  |
| --- |
| 1. private void button8\_Click(object sender, EventArgs e) 2. { 3. int rtn = 0; 4. string nameA = "/fruser/traj/A.txt"; 5. string nameB = "/fruser/traj/B.txt"; 6. rtn = robot.LoadTrajectoryLA(nameB, 0, 0, 0, 1, 100.0, 100.0, 1000.0); // 直线拟合 7. Console.WriteLine($"LoadTrajectoryLA rtn is {rtn}"); 8. DescPose startPos = new DescPose(0, 0, 0, 0, 0, 0); 9. robot.GetTrajectoryStartPose(nameA, ref startPos); 10. robot.MoveCart(startPos, 1, 0, (float)100.0, (float)100.0, (float)100.0, -1, -1); 11. rtn = robot.MoveTrajectoryLA(); 12. Console.WriteLine($"MoveTrajectoryLA rtn is {rtn}"); 13. } |

## 自定义碰撞检测阈值功能开始、结束

### 接口描述

章节：6、机器人安全设置 6.12 6.13章节

|  |
| --- |
| /\*\*   1. \* @brief 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 2. \* @param [in] 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 3. \* @param [in] jointDetectionThreshould 关节碰撞检测阈值 j1-j6 4. \* @param [in] tcpDetectionThreshould TCP碰撞检测阈值，xyzabc 5. \* @param [in] block 0-非阻塞；1-阻塞 6. \* @return 错误码 7. \*/ 8. int CustomCollisionDetectionStart(int flag, double[] jointDetectionThreshould, double[] tcpDetectionThreshould, int block) 9. /\*\* 10. \* @brief 自定义碰撞检测阈值功能关闭 11. \* @return 错误码 12. \*/ 13. int CustomCollisionDetectionEnd() |

### 代码示例

|  |
| --- |
| 1. private void button9\_Click(object sender, EventArgs e) 2. { 3. while (true) 4. { 5. int[] safety = { 5, 5, 5, 5, 5, 5 }; 6. robot.SetCollisionStrategy(3, 1000, 150, 250, safety); 7. double[] jointDetectionThreshold = { 0.3, 0.3, 0.3, 0.3, 0.3, 0.3 }; 8. double[] tcpDetectionThreshold = { 80, 80, 80, 80, 80, 80 }; 9. int rtn = robot.CustomCollisionDetectionStart(3, jointDetectionThreshold, tcpDetectionThreshold, 0); 10. Console.WriteLine($"CustomCollisionDetectionStart rtn is {rtn}"); 11. DescPose p1Desc = new DescPose(228.879, -503.594, 453.984, -175.580, 8.293, 171.267); 12. JointPos p1Joint = new JointPos(102.700, -85.333, 90.518, -102.365, -83.932, 22.134); 13. DescPose p2Desc = new DescPose(-333.302, -435.580, 449.866, -174.997, 2.017, 109.815); 14. JointPos p2Joint = new JointPos(41.862, -85.333, 90.526, -100.587, -90.014, 22.135); 15. ExaxisPos exaxisPos = new ExaxisPos(0.0, 0.0, 0.0, 0.0); 16. DescPose offdese = new DescPose(0.0, 0.0, 0.0, 0.0, 0.0, 0.0); 17. robot.MoveL(p1Joint, p1Desc, 0, 0, 100, 100, 100, -1, exaxisPos, 0, 0, offdese); 18. robot.MoveL(p2Joint, p2Desc, 0, 0, 100, 100, 100, -1, exaxisPos, 0, 0, offdese); 19. rtn = robot.CustomCollisionDetectionEnd(); 20. Console.WriteLine($"CustomCollisionDetectionEnd rtn is {rtn}"); 21. } 22. } |