

机器人C#SDK接口更新说明(3.8.1)

**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 |
| 2025/04/11 | V3.8.1 | 1. ConveyorSetParam()接口增加跟踪运动类型、跟踪起始距离、跟踪终止距离参数 2. 增加AccSmoothStart()接口 3. 增加AccSmoothEnd()接口 4. 增加RbLogDownload()接口 5. 增加AllDataSourceDownload()接口 6. 增加DataPackageDownload()接口 7. 增加GetRobotSN()接口 8. 增加ShutDownRobotOS()接口 9. 增加ConveyorComDetect()接口 10. 增加ConveyorComDetectTrigger()接口 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**说明：**

**①本手册适用于WebApp版本3.8.1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](#_Toc21396)

[1.1 传动带参数配置 1](#_Toc30196)

[1.1.1 接口描述 1](#_Toc12281)

[2 新增接口 1](#_Toc29993)

[2.1 加速度平滑开启 1](#_Toc1999)

[2.1.1 接口描述 1](#_Toc13676)

[2.2 加速度平滑关闭 2](#_Toc5974)

[2.2.1 接口描述 2](#_Toc7944)

[2.2.2 代码示例 2](#_Toc30351)

[2.3 控制器日志下载 2](#_Toc32595)

[2.3.1 接口描述 2](#_Toc25964)

[2.3.2 代码示例 3](#_Toc26273)

[2.4 所有数据源下载 3](#_Toc27252)

[2.4.1 接口描述 3](#_Toc8921)

[2.4.2 代码示例 3](#_Toc24448)

[2.5 数据备份包下载 4](#_Toc9141)

[2.5.1 接口描述 4](#_Toc9607)

[2.5.2 代码示例 4](#_Toc15856)

[2.6 获取控制箱SN码 4](#_Toc16266)

[2.6.1 接口描述 4](#_Toc11501)

[2.6.2 代码示例 4](#_Toc22048)

[2.7 关闭机器人操作系统 5](#_Toc28323)

[2.7.1 接口描述 5](#_Toc28728)

[2.7.2 代码示例 5](#_Toc13259)

[2.8 传送带通讯输入检测 5](#_Toc21255)

[2.8.1 接口描述 5](#_Toc11945)

[2.9 传送带通讯输入检测触发 6](#_Toc18998)

[2.9.1 接口描述 6](#_Toc7393)

[2.9.2 代码示例 6](#_Toc25305)

# 参数更改接口

## 传动带参数配置

### 接口描述

章节：14、其他接口

1. /\*\*
2. \* @brief 传动带参数配置
3. \* @param [in] para[0] 编码器通道 1~2
4. \* @param [in] para[1] 编码器转一圈的脉冲数
5. \* @param [in] para[2] 编码器转一圈传送带行走距离
6. \* @param [in] para[3] 工件坐标系编号 针对跟踪运动功能选择工件坐标系编号，跟踪抓取、TPD跟踪设为0
7. \* @param [in] para[4] 是否配视觉 0 不配 1 配
8. \* @param [in] para[5] 速度比 针对传送带跟踪抓取选项（1-100） 其他选项默认为1
9. \* @param [in] followType 跟踪运动类型，0-跟踪运动；1-追检运动
10. \* @param [in] startDis 追检抓取需要设置， 跟踪起始距离， -1：自动计算(工件到达机器人下方后自动追检)，单位mm， 默认值0
11. \* @param [in] endDis 追检抓取需要设置，跟踪终止距离， 单位mm， 默认值100
12. \* @return 错误码
13. \*/
14. int ConveyorSetParam(int encChannel, int resolution, double lead, int wpAxis, int vision, double speedRadio, int followType, int startDis, int endDis)

# 新增接口

## 加速度平滑开启

### 接口描述

章节：3. 机器人运动

1. /\*\*
2. \* @brief 加速度平滑开启
3. \* @param [in] saveFlag 是否断电保存
4. \* @return 错误码
5. \*/
6. int AccSmoothStart(bool saveFlag)

## 加速度平滑关闭

### 接口描述

章节：3. 机器人运动

1. /\*\*
2. \* @brief 加速度平滑关闭
3. \* @param [in] saveFlag 是否断电保存
4. \* @return 错误码
5. \*/
6. int AccSmoothEnd(bool saveFlag)

### 代码示例

1. private void button6\_Click(object sender, EventArgse)
2. {
3. bool saveFlag = false;
4. int rtn = 0;
5. JointPos p1Joint = new JointPos(88.927, -85.834, 80.289, -85.561, -91.388, 108.718);
6. DescPose p1Desc = new DescPose(88.739, -527.617, 514.939, -179.039, 1.494, 70.209);
7. JointPos p2Joint = new JointPos(27.036, -83.909, 80.284, -85.579, -90.027, 108.604);
8. DescPose p2Desc = new DescPose(-433.125, -334.428, 497.139, -179.723, -0.745, 8.437);
9. JointPos p3Joint = new JointPos(60.219, -94.324, 62.906, -62.005, -87.159, 108.598);
10. DescPose p3Desc = new DescPose(-112.215, -409.323, 686.497, 176.217, 2.338, 41.625);
11. ExaxisPos exaxisPos = new ExaxisPos(0, 0, 0, 0);
12. DescPose offdese = new DescPose(0, 0, 0, 0, 0, 0);
13. }

## 控制器日志下载

### 接口描述

章节：14、其他接口

1. /\*\*
2. \* @brief 控制器日志下载
3. \* @param savePath 保存文件路径"D://zDown/"
4. \* @return 错误码
5. \*/
6. int RbLogDownload(String savePath)

### 代码示例

1. private void button6\_Click(object sender, EventArgse)
2. {
3. Console.WriteLine("RbLogDownload start");
4. int rtn = robot.RbLogDownload(@"D:\zDOWN1\");
5. Console.WriteLine($"RbLogDownload rtn is {rtn}");
6. }

## 所有数据源下载

### 接口描述

章节：14、其他接口

1. /\*\*
2. \* @brief 所有数据源下载
3. \* @param savePath 保存文件路径"D://zDown/"
4. \* @return 错误码
5. \*/
6. int AllDataSourceDownload(String savePath)

### 代码示例

1. private void button6\_Click(object sender, EventArgse)
2. {
3. Console.WriteLine("AllDataSourceDownload start");
4. int rtn = robot.AllDataSourceDownload(@"D:\zDOWN\");
5. Console.WriteLine($"AllDataSourceDownload rtn is {rtn}");
6. }

## 数据备份包下载

### 接口描述

章节：14、其他接口

1. /\*\*
2. \* @brief 数据备份包下载
3. \* @param savePath 保存文件路径"D://zDown/"
4. \* @return 错误码
5. \*/
6. int DataPackageDownload(String savePath)

### 代码示例

1. private void button6\_Click(object sender, EventArgse)
2. {
3. Console.WriteLine("DataPackageDownload start");
4. int rtn = robot.DataPackageDownload(@"D:\zDOWN\");
5. Console.WriteLine($"DataPackageDownload rtn is {rtn}");
6. }

## 获取控制箱SN码

### 接口描述

章节：2、机器人基础

1. /\*\*
2. \* @brief 获取控制箱SN码
3. \* @param SNCode 控制箱SN码
4. \* @return 错误码
5. \*/
6. int GetRobotSN(ref string SNCode)

### 代码示例

1. private void button6\_Click(object sender, EventArgse)
2. {
3. string SN = "";
4. int rtn = robot.GetRobotSN(ref SN);
5. Console.WriteLine($"robot SN is {SN}");
6. }

## 关闭机器人操作系统

### 接口描述

章节：2、机器人基础

1. /\*\*
2. \* @brief 关闭机器人操作系统
3. \* @return 错误码
4. \*/
5. int ShutDownRobotOS()

### 代码示例

1. private void button6\_Click(object sender, EventArgse)
2. {
3. int rtn = robot.ShutDownRobotOS();
4. Console.WriteLine($"ShutDownRobotOS rtn is {rtn}");
5. }

## 传送带通讯输入检测

### 接口描述

章节：14、其他接口

1. /\*\*
2. \* @brief 传送带通讯输入检测触发
3. \* @return 错误码
4. \*/
5. int ConveyorComDetectTrigger()

## 传送带通讯输入检测触发

### 接口描述

章节：14、其他接口

1. /\*\*
2. \* @brief 传送带通讯输入检测触发
3. \* @return 错误码
4. \*/
5. int ConveyorComDetectTrigger()
   * 1. **代码示例**
6. private void button3\_Click(object sender, EventArgs e)
7. {
8. // 禁用按钮防止重复点击
9. button3.Enabled = false;
10. // 在后台线程中执行耗时操作
11. Thread conveyorThread = new Thread(ConveyorTest);
12. conveyorThread.IsBackground = true;
13. conveyorThread.Start();
14. }
15. private void button4\_Click(object sender, EventArgs e)
16. {
17. // 获取用户输入
18. string input = texBox.Text;
19. Console.WriteLine($"please input a number to trigger:{input}");
20. int rtn = robot.ConveyorComDetectTrigger();
21. Console.WriteLine($"ConveyorComDetectTrigger 返回值: {rtn}");
22. }
23. private void ConveyorTest()
24. {
25. // 使用Invoke来更新UI线程上的控件
26. this.Invoke((MethodInvoker)delegate {
27. Console.WriteLine("开始传送带测试...");
28. });
29. int retval = 0;
30. int index = 1;
31. int max\_time = 30000;
32. bool block = false;
33. retval = 0;
34. /\* 传送带抓取流程 \*/
35. DescPose startdescPose = new DescPose(139.176f, 4.717f, 9.088f, -179.999f, -0.004f, -179.990f);
36. JointPos startjointPos = new JointPos(-34.129f, -88.062f, 97.839f, -99.780f, -90.003f, -34.140f);
37. DescPose homePose = new DescPose(139.177f, 4.717f, 69.084f, -180.000f, -0.004f, -179.989f);
38. JointPos homejointPos = new JointPos(-34.129f, -88.618f, 84.039f, -85.423f, -90.003f, -34.140f);
39. ExaxisPos exaxisPos = new ExaxisPos(0, 0, 0, 0);
40. DescPose offdese = new DescPose(0, 0, 0, 0, 0, 0);
41. // 移动到安全位置
42. retval = robot.MoveL(homejointPos, homePose, 1, 1, 100, 100, 100, -1, exaxisPos, 0, 0, offdese, 1, 1);
43. Console.WriteLine($"MoveL 到安全位置返回值: {retval}");
44. // 传送带检测
45. retval = robot.ConveryComDetect(1000 \* 10);
46. Console.WriteLine($"ConveyorComDetect 返回值: {retval}");
47. // 获取跟踪数据
48. retval = robot.ConveyorGetTrackData(2);
49. Console.WriteLine($"ConveyorGetTrackData 返回值: {retval}");
50. // 开始跟踪
51. retval = robot.ConveyorTrackStart(2);
52. Console.WriteLine($"ConveyorTrackStart 返回值: {retval}");
53. // 移动到起始位置
54. robot.MoveL(startjointPos, startdescPose, 1, 1, 100, 100, 100, -1, exaxisPos, 0, 0, offdese, 1, 1);
55. robot.MoveL(startjointPos, startdescPose, 1, 1, 100, 100, 100, -1, exaxisPos, 0, 0, offdese, 1, 1);
56. // 结束跟踪
57. retval = robot.ConveyorTrackEnd();
58. Console.WriteLine($"ConveyorTrackEnd 返回值: {retval}");
59. // 返回安全位置
60. robot.MoveL(homejointPos, homePose, 1, 1, 100, 100, 100, -1, exaxisPos, 0, 0, offdese, 1, 1);
61. this.Invoke((MethodInvoker)delegate {
62. Console.WriteLine("传送带测试完成!");
63. button3.Enabled = true;
64. });
65. }