



 hitbot@hitbot.cc
www.hitbot.cc

Z-Arm- API-LINUX-PYTHON

领先的轻量型协作机械臂提供商

主营: 工业机器人 / 协作机器人 / 电动夹爪 / 电缸模组

慧灵科技（深圳）有限公司
Huiling-tech Robotic Co., LTD

发布说明

日期	版本	发布说明
2018.07.03	1.0.0.20180703_release	首次发布
2018.08.04	1.0.0.20180814_release	更正 is_stop 返回值定义
2020.02.28	2.0.0.20180814_release	常规更新
2020.08.28	2.0.0.20180814_release	1, 添加 new_movej 先运动算法 2, 添加五轴扩展控制接口 3, 同步其他 windows 功能

目录

1、准备工作.....	5
2、初始化流程.....	5
3、SDK 接口.....	6
3.1 HitbotInterface 类:	6
3.1.1 HitbotInterface 成员变量:	6
3.1.1.1 机械臂当前状态:	6
3.1.1.2 电爪状态.....	7
3.1.1.3 编码器位置.....	7
3.1.2 HitbotInterface 成员函数:	8
3.1.2.1 net_port_initial.....	8
3.1.2.2 initial.....	8
3.1.2.3 get_scara_param.....	9
3.1.2.4 unlock_position.....	9
3.1.2.5 is_connect.....	10
3.1.2.6 get_joint_state.....	10
3.1.2.7 set_drag_teach.....	11
3.1.2.8 get_drag_teach.....	12
3.1.2.9 set_cooperation_fun_state.....	12
3.1.2.10 get_cooperation_fun_state.....	13
3.1.2.11 is_collision.....	13
3.1.2.12 stop_move(过时, 不推荐使用).....	14
3.1.2.13 joint_home.....	14
3.1.2.14 movel_xyz.....	14
3.1.2.15 movej_xyz(过时, 推荐使用 new_movej_xyz_lr).....	16
3.1.2.16 movej_angle(过时, 推荐使用 new_movej_xyz_lr).....	17
3.1.2.17 change_attitude.....	18
3.1.2.18 wait_stop.....	19
3.1.2.19 pause_move.....	19
3.1.2.20 resume_move.....	20
3.1.2.21 set_digital_out.....	20

3.1.2.22 get_digital_out.....	20
3.1.2.23 get_digital_in.....	21
3.1.2.24 set_efg_state.....	21
3.1.2.25 get_efg_state.....	22
3.2.2.26 movej_xyz_lr(过时，推荐使用 new_movej_xyz_lr).....	22
3.2.2.27 new_movej_xyz_lr.....	24
3.2.2.28 new_set_acc.....	26
3.2.2.29 j5_motor_zero.....	26
3.2.2.30 set_j5_motor_pos.....	27
3.2.2.31 get_j5_parameter.....	27
3.2.2.32 movej_j5.....	28
3.2.2.33 get_efg_state_dji.....	29
3.2.2.34 set_efg_state_dji.....	29
3.2.2.35 new_stop_move.....	29
3.2.2.36 get_encoder_coor.....	30
3.1.2.37 new_movej_angle.....	30

开发环境：win7 win10 python2/3 x86/x64

1、准备工作

- 复制 HitbotInterface.py 和 libsmall_scara_interface.so 、 libsmall_scara_interface.so.1 、 libsmall_scara_interface.so.1.0 、 libsmall_scara_interface.so.1.0.0 到源码目录；
- 工程中添加 from HitbotInterface import HitbotInterface；
- 部分开发环境下 net_port_initial 中 CDLL 库目录需要修改成绝对路径；

2、初始化流程

```
from HitbotInterface import HitbotInterface

robot_id = 24
robot = HitbotInterface(robot_id)
robot.net_port_initial()

ret = robot.is_connect()
while ret != 1:
    time.sleep(0.1)
    ret = robot.is_connect()
    print(ret)

ret = robot.initial(1, 210)
if ret == 1:
    print("robot initial successful")
else:
    print("robot initial failed")
```

3、SDK 接口

3.1 HitbotInterface 类:

类名	HitbotInterface
功能	申请内存和初始化网络服务器，控制机械臂。

3.1.1 HitbotInterface 成员变量:

3.1.1.1 机械臂当前状态:

调用 <code>get_scara_param</code> 更新	
变量定义	功能
<code>card_number</code>	机械臂 id 号
<code>x</code>	x 坐标 (mm)
<code>y</code>	y 坐标 (mm)
<code>z</code>	z 坐标 (mm)
<code>r</code>	轴 4 角度 (deg)
<code>angle1</code>	轴 1 角度 (deg)
<code>angle2</code>	轴 2 角度 (deg)
<code>communicate_success</code>	<ul style="list-style-type: none">True: 不在线False: 在线
<code>initial_finish</code>	<ul style="list-style-type: none">True: 已初始化False: 未初始化

move_flag	<ul style="list-style-type: none">• True: 正在运动• False: 静止状态
-----------	--

3.1.1.2 电爪状态

调用 <code>get_efg_state</code> 更新	
变量定义	功能
efg_type	<ul style="list-style-type: none">• 8 : 当前控制的电爪类型 EFG-8• 20: 当前控制的电爪类型 EFG-20• 20: 没有设置电爪类型
efg_distance:	<p><code>efg_type==8:</code></p> <ul style="list-style-type: none">• 255: 夹指正在运动 <p><code>efg_distance:</code></p> <ul style="list-style-type: none">• 0 : 夹爪处于张开状态• 1 : 夹爪处于闭合状态 <p><code>efg_type==20: 夹指的当前位置</code></p>

3.1.1.3 编码器位置

调用 <code>get_encoder_coor</code> 更新	
变量定义	功能
encoder_x	编码器 x 轴坐标
encoder_y	编码器 y 轴坐标
encoder_z	编码器 z 轴坐标

<code>encoder_r</code>	编码器 r 轴坐标
<code>encoder_angle1</code>	编码器 angle1 轴坐标
<code>encoder_angle2</code>	编码器 angle2 轴坐标

3.1.2 HitbotInterface 成员函数:

3.1.2.1 `net_port_initial`

功能	申请内存和初始化网络服务器
注意	调用其他 api 之前，需要先调用该函数，否则将导致程序崩溃，同一个 python 程序控制多台手臂时，每一个手臂对象都要调用改函数
函数定义	<code>def net_port_initial(self)</code>
参数	无
返回	<ul style="list-style-type: none"> • 1 成功 • 0 失败，一般是 40000 端口号被占用

3.1.2.2 `initial`

功能	初始化机械臂的各项参数，机械臂进入工作状态
注意	<ul style="list-style-type: none"> • 机械臂在运动时调用该函数，会造成机械臂抖动 • 返回 3 时，可以使用 <code>joint_home</code> 使关节强制回零
函数定义	<code>def initial(self, generation, z_trail)</code>
参数	<ul style="list-style-type: none"> • <code>int generation</code>: 400mm 臂展系列传入 1, 320mm 臂展系列

	<p>及其他臂展手臂传入 5,</p> <ul style="list-style-type: none">• <code>float z_travel</code>: 上下关节有效行程 (mm)
返回	<ul style="list-style-type: none">• 0: 机械臂不在线• 1: 初始化成功• 2: <code>generation</code> 参数错误• 3: 机械臂当前位置不在限定范围, 可以使用 <code>joint_home</code> 使关节强制回零• 12: <code>z_travel</code> 传参错误• 101: 传入参数 NOT A NUMBER• 105: 存在某一个关节失效• >= 10000, pid 自检异常

3.1.2.3 `get_scara_param`

功能	更新 x, y, z 等成员变量
注意	机械臂不在线时, 坐标参数无意义
函数定义	<code>def get_scara_param(self)</code>
参数	无
返回	无

3.1.2.4 `unlock_position`

功能	解锁机械臂, 使机械臂可以接受运动指令
注意	一般初始化成功后立刻调用

函数定义	<code>def unlock_position(self)</code>
参数	无
返回	<ul style="list-style-type: none">• 0: 机械臂不在线• 1: 成功

3.1.2.5 `is_connect`

功能	查询机械臂是否在线
注意	无
函数定义	<code>def is_connect(self)</code>
参数	无
返回	<ul style="list-style-type: none">• False: 机械臂不在线• True: 机械臂在线

3.1.2.6 `get_joint_state`

功能	获取机械臂关节状态
注意	无
函数定义	<code>def get_joint_state(self, joint_num)</code>
参数	<code>int joint_num</code> : 轴号 1-4
返回	<ul style="list-style-type: none">• 0: 轴发生复位, 需要重新初始化• 1: 关节正常• 2: 传入参数超范围

- 3: 未初始化
- 4: 轴状态获取失败
- 5: 发生碰撞;
- 6: 处于拖动模式
- 7: 关节缺轴/失效
- 8: 编码器错误
- 9: 电机堵转/过电流保护
- 10: 过压保护
- 11: 手臂收到 PC 发送的错误数据
- 12: 手臂主控与驱动通讯异常
- 21: MOS ntc 开路
- 22: MOS ntc 短路
- 23: 电机 ntc 开路
- 24: 电机 ntc 短路
- 25: MOS ntc 温度过高
- 26: MOS ntc 温度过低
- 27: 电机 ntc 温度过高
- 28: 电机 ntc 温度过低

3.1.2.7 `set_drag_teach`

功能	设定拖动示教功能是否开启
----	--------------

注意	<ul style="list-style-type: none">仅协作机型支持开启此功能拖动示教功能开启后，轴 3 不支持拖动，需要通过调用运动函数控制轴 3，部分新机型支持 z 轴拖动示教
函数定义	<code>def set_drag_teach(self, enable)</code>
参数	<code>bool enable</code> : True 开启, False 关闭

3.1.2.8 `get_drag_teach`

功能	查询是否处于拖动示教状态
注意	无
函数定义	<code>def get_drag_teach(self)</code>
参数	无

3.1.2.9 `set_cooperation_fun_state`

功能	设定协作功能是否开启，开启后机械臂遇到障碍物会停止运动，并上报此状态
注意	仅协作机型支持开启此功能

函数定义	<code>def set_cooperation_fun_state(self, enable)</code>
参数	<code>bool enable</code> : True 开启, False 关闭
返回	<ul style="list-style-type: none">• False: 成功• True: 失败

3.1.2.10 `get_cooperation_fun_state`

功能	查询机械臂是否开启协作功能
注意	
函数定义	<code>def get_cooperation_fun_state(self)</code>
参数	无
返回	<ul style="list-style-type: none">• False: 开启• True: 关闭

3.1.2.11 `is_collision`

功能	查询机械臂是否发生碰撞
注意	无
函数定义	<code>def is_collision(self)</code>
参数	无
返回	<ul style="list-style-type: none">• False: 未发生碰撞• True: 发生碰撞

3.1.2.12 `stop_move`(过时, 不推荐使用)

功能	立即结束当前正在执行的指令
注意	无
函数定义	<code>def stop_move(self)</code>
参数	无
返回	无

3.1.2.13 `joint_home`

功能	使轴回到零位
注意	无
函数定义	<code>def joint_home(self,joint_num)</code>
参数	<ul style="list-style-type: none">• <code>int joint_num</code>: 轴号
返回	<ul style="list-style-type: none">• 0: 未连接• 1: 调用成功• 2: 传入参数错误• 3: 机械臂正在初始

3.1.2.14 `movel_xyz`

功能	以 <code>movel</code> 的运动方式到达目标点
注意	<ul style="list-style-type: none">• <code>move_flag</code> 等于 <code>False</code> 时, 进行首次调用, 返回 1 后, 可以重复调用该函数设定新的目标点位

	<ul style="list-style-type: none">距离当前目标点有一定距离时，再次设定新的目标，机械臂将以一定的速度通过当前目标点，而后去往新的目标点。
函数定义	<pre>def movel_xyz(self, goal_x, goal_y, goal_z, goal_r, speed)</pre>
参数	<ul style="list-style-type: none"><code>float goal_x</code>: 目标点 x 坐标 (mm)<code>float goal_y</code>: 目标点 y 坐标 (mm)<code>float goal_z</code>: 目标点 z 坐标 (mm)<code>float goal_r</code>: 目标点 r 坐标 (deg)<code>float speed</code>: xyz 点的线速度 (mm/s) 或轴 4 的旋转角速度 (deg/s)
返回	<ul style="list-style-type: none">0: 正在执行其他指令，本次指令无效1: 本次指令生效，机械臂开始运动2: 设置速度小于等于零3: 未初始化4: 过程点无法到达6: 伺服未开启7: 存在中间过程点无法以机械臂当前姿态（手系）达到11: 手机端在控制99: 急停中101: 传入参数 NOT A NUMBER102: 发生碰撞，本次指令无效103: 轴发生复位，需要重新初始化，本次指令无效

3.1.2.15 movej_xyz(过时, 推荐使用 new_movej_xyz_lr)

功能	以 movej 的运动方式到达目标点
注意	<ul style="list-style-type: none">move_flag 等于 False 时, 进行首次调用, 返回 1 后, 可以重复调用该函数设定新的目标点位距离当前目标点有一定距离时, 并且当前目标点 roughly 大于 0, 再次设定新的目标, 机械臂将以一定的速度在当前目标点附近通过, 而后去往新的目标点。
函数定义	<code>def movej_xyz(self, goal_x, goal_y, goal_z, goal_r, speed, roughly)</code>
参数	<ul style="list-style-type: none"><code>float goal_x</code>: 目标点 x 坐标 (mm)<code>float goal_y</code>: 目标点 y 坐标 (mm)<code>float goal_z</code>: 目标点 z 坐标 (mm)<code>float goal_r</code>: 目标点 r 坐标 (deg)<code>float speed</code>: xyz 点的线速度 (mm/s) 或轴 4 的旋转角速度 (deg/s)<code>float roughly</code>: 0 先运动到前目标点, 并且在当前目标点的速度等于 0, 而后在去往新的目标点, 1 在有新目标点时, 将在当前目标点附近以最大速度通过, 此参数取值范围 0 到 1, 越大通过速度越快, 但是通过点距离当前目标点越远。
返回	<ul style="list-style-type: none">-2: 不支持该运动函数0: 正在执行其他指令, 本次指令无效1: 本次指令生效, 机械臂开始运动

- 2: 设置速度小于等于零
- 3: 未初始化
- 4: 目标点无法到达
- 6: 伺服未开启
- 11: 手机端在控制
- 101: 传入参数 NOT A NUMBER
- 102: 发生碰撞，本次指令无效
- 103: 轴发生复位，需要重新初始化，本次指令无效

3.1.2.16 movej_angle(过时，推荐使用 new_movej_xyz_lr)

功能	以 movej 的运动方式到达目标点
注意	<ul style="list-style-type: none">• move_flag 等于 False 时，进行首次调用，返回 1 后，可以重复调用该函数设定新的目标点位• 距离当前目标点有一定距离时，并且当前目标点 roughly 大于 0，再次设定新的目标，机械臂将以一定的速度在当前目标点附近通过，而后去往新的目标点。
函数定义	<pre>def movej_angle(self, goal_angle1, goal_angle2, goal_z, goal_r, speed, roughly)</pre>
参数	<ul style="list-style-type: none">• float goal_angle1: 目标点轴 1 角度坐标 (deg)• float goal_angle2: 目标点轴 2 角度坐标 (deg)• float goal_z: 目标点 z 坐标 (mm)• float goal_r: 目标点 r 坐标 (deg)

	<ul style="list-style-type: none">• float speed: xyz 点的线速度 (mm/s) 或轴 4 的旋转角速度 (deg/s)• float roughly: 0 先运动到前目标点，并且在当前目标点的速度等于 0，而后在去往新的目标点，1 在有新目标点时，将在当前目标点附近以最大速度通过，此参数取值范围 0 到 1，越大通过速度越快，但是通过点距离当前目标点越远。
返回	<ul style="list-style-type: none">• -2: 不支持该运动函数• 0: 正在执行其他指令，本次指令无效• 1: 本次指令生效，机械臂开始运动• 2: 设置速度小于等于零• 3: 未初始化• 4: 目标点无法到达• 6: 伺服未开启• 11: 手机端在控制• 101: 传入参数 NOT A NUMBER• 102: 发生碰撞，本次指令无效• 103: 轴发生复位，需要重新初始化，本次指令无效

3.1.2.17 change_attitude

功能	切换手系
注意	
函数定义	<code>def change_attitude(self, speed)</code>

参数	float speed: 线速度 (mm/s)
----	-------------------------

3.1.2.18 [wait_stop](#)

功能	用于判断运动是否结束
注意	<ul style="list-style-type: none">• wait_stop 属于开环控制，函数返回以后不代表手臂一定处于静止状态，在关键点位需要搭配 is_robot_goto_target 使用，此时 is_robot_goto_target 需要做超时处理，一般超时时间 3s 即可• 阻塞函数，运动数据下发完成后返回
函数定义	<code>def wait_stop(self):</code>
参数	无
返回	<ul style="list-style-type: none">• true: 正常停止• false: 存在某些异常使 wait_stop 超时,此时应停止手臂作业流程

3.1.2.19 [pause_move](#)

功能	暂停机械臂的运动
注意	函数返回后，经过一定延时（几百毫秒），机械臂停止运动
函数定义	<code>def pause_move(self):</code>
参数	无
返回	<ul style="list-style-type: none">• 默认返回 1

3.1.2.20 resume_move

功能	恢复机械臂的运动
注意	无
函数定义	<code>def resume_move(self)</code>
参数	无
返回	<ul style="list-style-type: none">默认返回 1

3.1.2.21 set_digital_out

功能	设定 io 输出点状态
注意	
函数定义	<code>def set_digital_out(self, io_number, io_value)</code>
参数	<ul style="list-style-type: none"><code>int io_number</code>: io 输出点编号<code>bool value</code>: 设定值
返回	<ul style="list-style-type: none">0: <code>io_number</code> 参数错误1: 设置成功3: 未初始化;

3.1.2.22 get_digital_out

功能	获取 io 输出点状态
注意	
函数定义	<code>def get_digital_out(self, io_number)</code>

参数	<code>int io_number</code> : io 输出点编号
返回	<ul style="list-style-type: none">• -1: <code>io_number</code> 参数错误• 0: io 口输出状态为断开• 1: io 口输出状态为导通• 3: 未初始化

3.1.2.23 `get_digital_in`

功能	获取 io 输入点状态
注意	
函数定义	<code>def get_digital_in(self, io_number)</code>
参数	<code>int io_number</code> : io 输入点编号
返回	<ul style="list-style-type: none">• -1: <code>io_number</code> 参数错误• 0: io 输入点没有被触发• 1: io 输入点被触发• 3: 未初始化

3.1.2.24 `set_efg_state`

功能	设定电动夹爪状态
注意	切换控制类型需要重启机械臂和 <code>sdk</code>
函数定义	<code>def set_efg_state(self, efg_type, efg_distance)</code>
参数	<ul style="list-style-type: none">• <code>int type</code>: 8 EFG-8, 20 EFG-20

	<ul style="list-style-type: none">• float <code>distance</code>:
	Type==8:
	0 张开
	1 闭合
	Type==20: 实际行程 (mm)
返回	<ul style="list-style-type: none">• -1: 控制参数发生变化• 0: type 参数错误• 1: 设置成功• 3: 未初始化;

3.1.2.25 `get_efg_state`

功能	获取电动夹爪状态
注意	<ul style="list-style-type: none">• 返回 1, 更新成员变量中的 <code>efg_type</code> 和 <code>efg_distance</code>• 1632 机型不支持在运动过程中调用
函数定义	<code>def get_efg_state(self)</code>
参数	无
返回	<ul style="list-style-type: none">• 1: 成功• 3: 失败, 未初始化;

3.2.2.26 `movej_xyz_lr`(过时, 推荐使用 `new_movej_xyz_lr`)

功能	以 movej 的运动方式, 并以指定手系达目标点
----	---------------------------

	<ul style="list-style-type: none">• move_flag 等于 false 时，进行首次调用，返回 1 后，可以重复调用该函数设定新的目标点位
注意	<ul style="list-style-type: none">• 距离当前目标点有一定距离时，并且当前目标点 roughly 大于 0，再次设定新的目标，机械臂将以一定的速度在当前目标点附近通过，而后去往新的目标点。
函数定义	<pre>def movej_xyz_lr(self, goal_x, goal_y, goal_z, goal_r, speed, roughly, lr)</pre>
参数	<ul style="list-style-type: none">• float goal_x: 目标点 x 坐标 (mm)• float goal_y: 目标点 y 坐标 (mm)• float goal_z: 目标点 z 坐标 (mm)• float goal_r: 目标点 r 坐标 (deg)• float speed: xyz 点的线速度 (mm/s) 或轴 4 的旋转角速度 (deg/s)• float roughly: 0 先运动到前目标点，并且在当前目标点的速度等于 0，而后在去往新的目标点，1 在有新目标点时，将在当前目标点附近以最大速度通过，此参数取值范围 0 到 1，越大通过速度越快，但是通过点距离当前目标点越远。• int lr: 1 右手系，-1 左手洗，angle2>0 右手系，否则为左手系
返回	<ul style="list-style-type: none">• -2: 不支持该运动函数• 0: 正在执行其他指令，本次指令无效• 1: 本次指令生效，机械臂开始运动• 2: 设置速度小于等于零

	<ul style="list-style-type: none">• 3: 未初始化• 4: 目标点无法到达• 6: 伺服未开启• 7: 无法以设定手系到达目标点位• 11: 手机端在控制• 99: 急停中• 101: 传入参数 NOT A NUMBER• 102: 发生碰撞, 本次指令无效• 103: 轴发生复位, 需要重新初始化, 本次指令无效
--	--

3.2.2.27 new_movej_xyz_lr

功能	以 movej 的运动方式, 并以指定手系达目标点
注意	<ul style="list-style-type: none">• move_flag 等于 false 时, 进行首次调用, 返回 1 后, 可以重复调用该函数设定新的目标点位• 距离当前目标点有一定距离时, 并且当前目标点 roughly 大于 0, 再次设定新的目标, 机械臂将以一定的速度在当前目标点附近通过, 而后去往新的目标点。
函数定义	<pre>def new_movej_xyz_lr(self,goal_x,goal_y,goal_z,goal_r,speed,roughly,lr)</pre>
参数	<ul style="list-style-type: none">• float goal_x: 目标点 x 坐标 (mm)• float goal_y: 目标点 y 坐标 (mm)• float goal_z: 目标点 z 坐标 (mm)

	<ul style="list-style-type: none">• float <code>goal_r</code>: 目标点 r 坐标 (deg)• float <code>speed</code>: xyz 点的线速度 (mm/s) 或轴 4 的旋转角速度 (deg/s)• float <code>roughly</code>: 0 先运动到前目标点，并且在当前目标点的速度等于 0，而后在去往新的目标点，1 在有新目标点时，将在当前目标点附近以最大速度通过，此参数取值范围 0 到 1，越大通过速度越快，但是通过点距离当前目标点越远。• int <code>lr</code>: 1 右手系，-1 左手洗，angle2>0 右手系，否则为左手系
返回	<ul style="list-style-type: none">• 0: 正在执行其他指令，本次指令无效• 1: 本次指令生效，机械臂开始运动• 2: 设置速度小于等于零• 3: 未初始化• 4: 目标点无法到达• 6: 伺服未开启• 7: 无法以设定手系到达目标点位• 11: 手机端在控制• 99: 急停中• 101: 传入参数 NOT A NUMBER• 102: 发生碰撞，本次指令无效• 103: 轴发生复位，需要重新初始化，本次指令无效

3.2.2.28 new_set_acc

功能	设定 new_movej_xyz_lr 运动模型的关节加速度百分比
注意	无, 范围>=30,<=220
函数定义	<pre>def new_set_acc(self,j1_max_acc, j2_max_acc, j3_max_acc, j4_max_acc)</pre>
参数	<ul style="list-style-type: none">int j1_max_acc: 关节 1 加速度百分比int j2_max_acc: 关节 2 加速度百分比int j3_max_acc: 关节 3 加速度百分比int j4_max_acc: 关节 4 加速度百分比
返回	<ul style="list-style-type: none">1: 默认返回 1

3.2.2.29 j5_motor_zero

功能	1) 搭配步进电机扩展模块使用, 调用后会将当前位置作为零位 2) 搭配第五轴使用时, 无效
注意	无
函数定义	<pre>def j5_motor_zero()</pre>
参数	无
返回	<ul style="list-style-type: none">0: 手臂未连接1: 设定成功3: 手臂未初始化4: 不支持该功能

3.2.2.30 set_j5_motor_pos

功能	1) 搭配步进电机扩展模块使用, 设定步进电机的当前位置, 范围限制正负 1000 圈 2) 搭配第五轴使用时, 范围正负 150 度
注意	手臂本体四个关节在运动时, 禁止调用此接口
函数定义	<code>def set_j5_motor_pos(self,deg,speed):</code>
参数	<ul style="list-style-type: none">• <code>float deg</code>: 目标角度 单位 deg• <code>float speed</code>: 平均速度 单位 deg/s
返回	<ul style="list-style-type: none">• 0: 其他运动函数调用中• 1: 设定成功• 2: speed 小于等于零• 3: 未初始化• 4: 手臂当前 xyzr 超出运动范围• 99: 急停中• 102: 手臂发生碰撞• 103: 关节复位• 105: 关节异常• 106: 不支持该功能

3.2.2.31 get_j5_parameter

功能	搭配步进电机扩展模块或第五轴使用, 返回电机电机当前位置
----	------------------------------

注意	无
函数定义	<code>def get_j5_parameter():</code>
参数	无
返回	<ul style="list-style-type: none">返回五轴当前位置

3.2.2.32 movej_j5

功能	搭配第五轴使用
注意	第五轴旋转范围正负 150deg
函数定义	<code>def movej_j5(self, j5_pos, speed):</code>
参数	<ul style="list-style-type: none"><code>float deg</code>: 目标角度 单位 deg<code>float speed</code>: 平均速度 单位 deg/s
返回	<ul style="list-style-type: none">0: 其他运动函数调用中1: 设定成功2: speed 小于等于零3: 未初始化4: 手臂当前 xyzr 超出运动范围99: 急停中102: 手臂发生碰撞103: 关节复位105: 关节异常106: 不支持该功能

3.2.2.33 `get_efg_state_dji`

功能	搭配 EFG-20 上电夹指不动版本使用，用于查询夹爪位置
注意	仅用于控制上电夹指不动 EFG-20 电动夹爪，上电位置为 0
函数定义	<code>def get_efg_state_dji(self)</code>
参数	无
返回	<ul style="list-style-type: none">返回夹爪当前类型和夹指当前位置

3.2.2.34 `set_efg_state_dji`

功能	搭配 EFG-20 上电夹指不动版本使用，用于控制夹爪移动
注意	仅用于控制上电夹指不动 EFG-20 电动夹爪，上电位置为 0
函数定义	<code>def set_efg_state_dji(self, type, distance)</code>
参数	<ul style="list-style-type: none"><code>int type</code>: 传入 20<code>float distance</code>: type==20: 实际行程 (mm), 范围正负 40
返回	<ul style="list-style-type: none">-1: 控制参数发生变化0: type 参数错误1: 设置成功3: 未初始化;

3.2.2.35 `new_stop_move`

功能	立即结束目前正在执行的指令
注意	相比 <code>stop_move</code> ，优化了停止算法，建议搭配 <code>new_movej_xyz_lr</code> 使用

函数定义	<code>def new_stop_move()</code>
参数	无
返回	无

3.2.2.36 `get_encoder_coor`

功能	更新编码器位置，调用后 <code>encoder_x, encoder_y, encoder_z</code> 等变量会进行更新
注意	
函数定义	<code>def get_encoder_coor(self)</code>
参数	无
返回	无

3.1.2.37 `new_movej_angle`

功能	以 movej 的运动方式到达目标点
注意	<ul style="list-style-type: none"> • <code>move_flag</code> 等于 <code>False</code> 时，进行首次调用，返回 1 后，可以重复调用该函数设定新的目标点位 • 距离当前目标点有一定距离时，并且当前目标点 <code>roughly</code> 大于 0，再次设定新的目标，机械臂将以一定的速度在当前目标点附近通过，而后去往新的目标点。
函数定义	<code>def new_movej_angle(self, goal_angle1, goal_angle2, goal_z, goal_r, speed, roughly)</code>
参数	<ul style="list-style-type: none"> • <code>float goal_angle1</code>: 目标点轴 1 角度坐标 (deg) • <code>float goal_angle2</code>: 目标点轴 2 角度坐标 (deg)

- `float goal_z`: 目标点 z 坐标 (mm)
- `float goal_r`: 目标点 r 坐标 (deg)
- `float speed`: xyz 点的线速度 (mm/s) 或轴 4 的旋转角速度 (deg/s)
- `float roughly`: 0 先运动到前目标点，并且在当前目标点的速度等于 0，而后在去往新的目标点，1 在有新目标点时，将在当前目标点附近以最大速度通过，此参数取值范围 0 或 1

• 0: 正在执行其他指令，本次指令无效

• 1: 本次指令生效，机械臂开始运动

• 2: 设置速度小于等于零

• 3: 未初始化

• 4: 目标点无法到达

返回

• 6: 伺服未开启

• 11: 手机端在控制

• 99: 急停中

• 101: 传入参数 NOT A NUMBER

• 102: 发生碰撞，本次指令无效

• 103: 轴发生复位，需要重新初始化，本次指令无效



HITBOT

慧灵科技(深圳)有限公司
Huiling-tech Robotic Co.,Ltd

电话 0755-36382405
邮箱 hitbot@hitbot.cc
网址 www.hitbot.cc
地址 广东省深圳市宝安区西乡街道南昌社区航城大道
华丰国际机器人产业园B栋六层601-605

