JAKA Software Development Kit Documentation

Python

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd
电话 Tel ：+400 006 2665 | 网站 Web:www.jaka.com
上海：上海市闵行区东川路 555 号紫竹科技园 3 号楼 201 | Room 201, Building 3,No.555 Dongchuan Rd, Shanghai
常州：江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

Version record

Version number	Version date	The version of the controller is supported	Description
V1.0.0	2020.3.24	V1.4.10/V2.0.10	Create
V1.0.0	2020.6.24	V1.4.10/V2.0.10	Add additional instructions for different motion instructions
V1.0.14	2020.7.24	V1.4.10/V2.0.10	5.61 to 5.64 are new interfaces that modify jog_stop the interface

上海节卡机器人科技有限公司　Shanghai JAKA Robotics Ltd
电话 Tel ：+400 006 2665 | 网站 Web:www.jaka.com
上海：上海市闵行区东川路 555 号紫竹科技园 3 号楼 201 | Room 201, Building 3,No.555 Dongchuan Rd, Shanghai
常州：江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

Catalogue

1. Introduction

This document is a python version of the Software Development Kit document.

2. What to know about the documentation

- Running environment: linux python 3.5 32 bits,windows Python 3.7 64 bits

Units used to parameters: mm (mm), radian (rad).

In **non-specific code examples, the robot is turned on by default and powered on**

All code samples in the documentation default to no interference in the robot's workspace

The examples in the documentation are all default to the user's Python interpreter being able to find the jkrc module

2.1 Use under linux

Linux needs to libjakaAPI.so and jkrc.so under the same folder and add the current folder path to the environment variable,

export LD_LIBRARY_PATH/xx/xx/

2.2 Use under windows

Windows needs to put jkrc and jakaAPI .dll folder, and FAQs can be queried for frequently asked questions.

2.3 Dynamic library version number query method

The normal use of jkrc requires a dynamic library file, and here's how to query the dynamic library version number:

Right-click the dll file in Windows, select the properties, and you can see the version information in the Details tab.

Enter the command strings in **Linux libjakaAPI.so | Grep jakaAPI_version** query version information.

3. Data Types

3.1 IO type

JAKA robots have three types of IO, which are control cabinet IO, tool IO, and extended IO, respectively.

The following IO definitions are used in the following sample code:

```
IO_CABINET =0            # Control cabinet IO
IO_TOOL = 1              # tool IO
IO IO_EXTEND = 2         # extened IO
```

3.2 Coordinate system type

JAKA robot has three coordinate systems, namely the base coordinate system / the current user coordinate system, joint space, and the tool coordinate system, respectively. The following coordinate system definitions are used in the following sample code:

```
COORD_BASE = 0          # the base coordinate system / the  current user coordinate system
COORD_JOINT = 1         # joint space
COORD_TOOL = 2          # the tool coordinate system
```

3.3 Sport type

There are two types of JAKA robot motion, which are defined as follows:

```
ABS = 0                 # Absolute move
INCR = 1                # Incremental move
```

4. Return values

Type	Returns a value	Describe
ERR_SUCC	0	Success
ERR_INVALID_HANDLER	-1	An invalid handle
ERR_INVALID_PARAMETER	-2	An invalid argument
ERR_COMMUNICATION_ERR	-3	There was a communication error
ERR_KINE_INVERSE_ERR	-4	The reverse solution failed
ERR_EMERGENCY_PRESSED	-5	The emergency stop key has not been released
ERR_NOT_POWERED	-6	The robot is not powered on
ERR_NOT_ENABLED	-7	The robot is not enabled
ERR_DISABLE_SERVOMODE	-8	SERVOMODE mode is not entered
ERR_NOT_OFF_ENABLE	-9	The power is not lowered before power is turned off
ERR_PROGRAM_IS_RUNNING	-10	The program is running
ERR_CANNOT_OPEN_FILE	-11	Opening the file failed

All return values of functions are a tuple except the function RC(ip)（as seen in 5.1 Instantiated robots）. The return values of the Get class's functions are in the form of **(errcode, data),** the first element is the error code. When the robot is working well，the errcode is equal to 0. Otherwise, the errcode value is not equal to 0, and the value can be queried in the table above. The second element is data, such as joint angle values, and so on.

For example, the function '**get_joint_position()**' is utilized to get the the joint angles. The return value is (0, [1 2 3 4 5 6]), which means that the joint angles is sucessfully obtained. The return value is (-3,0), which indicates the joint angles cannot be obtained due to communication errors.

5. Functions

5.1 Instantiated robots

Function	RC(ip)
Describe	Instantize a robot object

Parameters	ip: The robot's IP address needs to be filled in with a string only the correct IP address instantiated object to control the robot.
Returns a value	Success: Returning a robot object Failed: The created object is destroyed

Example:

```
import jkrc
robot = jkrc.RC("192.168.2.64")  # Instantizing a robot object
```

5.2 login in

Function	**login()**
Describe	Connect the robot controller
Parameters	
Returns a value	success(0,) failed:others

5.3 Logout

Function	**logout()**
Describe	Disconnect the controller
Parameters	
Returns a value	Success :(0,) Failed: Other

Example:

```
import jkrc
robot =jkrc.RC("192.168.2.64")    # Instantizing a robot object
robot.login()                     #Connect the robot controller
pass
robot.logout()                    #Disconnect the controller
```

5.4 Get SDK Version No.

Function	**get_sdk_version()**
Describe	Get the SDK version number
Parameters	
Returns a value	Success: (0,version) failed: Other

Example:

```
import jkrc
robot = jkrc.RC("192.168.2.64")
robot.login()
ret = robot.get_sdk_version()
print("SDK version is:",ret[1])
robot.logout()
```

5.5 Power on

Function	power_on()
Describe	Turning on the robot and powering it on. The robot will have a delay of about 8 seconds
Parameters	
Returns a value	Success：(0,) failed: Other

Example:

```
import jkrc
robot = jkrc.RC("192.168.2.64")
robot.login()
robot.power_on()
robot.logout()
```

5.6 Power off

Function	power_off()
Describe	Turn off the robot
Parameters	
Returns a value	Success :(0,) Failed: Other

5.7 Shut down Robot controller

Function	shut_down()
Describe	The robot control cabinet shuts down
Parameters	
Returns a value	Success :(0,) Failed: Other

5.8 Enable the robot

Function	enable_robot()
Describe	Enable the robot
Parameters	
Returns a value	Success :(0,) Failed: Other

Example:

```
import jkrc
robot = jkrc.RC("192.168.2.64")
robot.login()
robot.enable_robot()
robot.logout()
```

5.9 Disable the robot

Function	disable_robot()
Describe	disable the robot
Parameters	
Returns a value	Success :(0,) Failed: Other

5.10 Control robot movement in manual mode

Function	Jog (aj_num , move_mode, coord_type, jog_vel, pos_cmd)
Describe	Control robot movement in manual mode
Parameters	aj_num： Represent joint number [0-5] in joint space, and x, y, z, rx, ry, rz-axis in Cartesian space move_mode： Robot move mode, incremental move(0) or absolute move(1) coord_type： Robot move coordinate frame, tool coordinate frame, base coordinate frame (current world/user coordinate frame) or joint space jog_vel： Command velocity, unit of rotation axis or joint move is rad/s, move axis unit is mm/s pos_cmd： Command position, unit of rotation axis or joint move is rad, move axis unit is mm
Returns a value	Success :(0,) Failed: Other

Example:

```
"jog  movement" ""
"1. Joint space jog" .'
import jkrc
import time
```

```
COORD_BASE = 0
COORD_JOINT = 1
COORD_TOOL = 2
ABS = 0
INCR = 1
joint1 = 0
joint2 = 1
joint3 = 2
joint4 = 3
joint5 = 4
joint6 = 5
robot = jkrc.RC("192.168.2.64")
robot.login()
robot.jog(aj_num = joint1 ,move_mode = INCR,coord_type = COORD_JOINT ,jog_vel=
5,pos_cmd = 0.5) robot.jog_stop()
robot.logout()
"'2. Cartesian spacejog'"
import jkrc
import time
COORD_BASE = 0
COORD_JOINT = 1
COORD_TOOL = 2
ABS = 0
INCR = 1
cart_x = 0
cart_y = 1
cart_z = 2
cart_rx = 3
cart_ry = 4
cart_rz = 5
robot = jkrc.RC("192.168.2.64")
 robot.login()
robot.jog(aj_num = cart_z ,move_mode = INCR,coord_type = COORD_BASE ,jog_vel=
5,pos_cmd = 10)
robot.jog_stop()
robot.logout()
```

Tips:

If the robot is already close to a singular posture, the jog robot will not be able to use the sample code above

5.11 Control robot movement stop in manual mode

Function	jog_stop(joint_num)
Describe	Stop the robot in manual mode
Parameters	joint_num: Robot axis number 0-5, when number is -1, stop all axes
Returns a value	Success :(0,) Failed: Other

```
import jkrc
import time
COORD_BASE = 0
COORD_JOINT = 1
COORD_TOOL = 2
ABS = 0
INCR = 1
joint1 = 0
joint2 = 1
joint3 = 2
joint4 = 3
joint5 = 4
joint6 = 5
robot = jkrc.RC("192.168.2.194")
 robot.login()
robot.jog(aj_num = joint1 ,move_mode = INCR,coord_type = COORD_JOINT ,jog_vel=
0.1,pos_cmd = -1.5)
time.sleep(2)
robot.jog_stop(-1)
robot.logout()
```

5.12 Get joint angle

Function	get_joint_position()
Description	Get the robot 6 joint values
Parameter	
Return	success：(0,joint_pos) joint_pos likes this: (j1,j2,j3,j4,j5,j6) ,j1 means the angle of joint 1 fail：other

```
import jkrc
```

```
robot = jkrc.RC("192.168.2.64")
robot.login()
ret = robot.get_joint_position() if ret[0] == 0:
print("the joint position is :",ret[1]) else:
print("some things happend,the errcode is: ",ret[0])
robot.logout()
```

5.13 Get tcp pose

Function	get_tcp_position()
Description	Get tcp pose
Parameter	
Return	success：(0, cartesian_pose) cartesian_pose is like (x,y,z,rx,ry,rz) fail：other

```
import jkrc
robot = jkrc.RC("192.168.2.64")
robot.login()
ret = robot.get_tcp_position() if ret[0] == 0:
print("the tcp position is :",ret[1]) else:
print("some things happend,the errcode is: ",ret[0]) robot.logout()
```

5.14 Kine inverse

Function	kine_inverse(ref_pos, cartesian_pose)
Description	Calculate the kine inverse of the specified pose under the current tool, current installation angle, and current user coordinate frame settings
Parameter	ref_pos: Reference joint position for kine inverse cartesian_pose: Cartesian space pose value
Return	success：(0, joint_pos) joint_pos is like (j1,j2,j3,j4,j5,j6) fail：other

5.15 Kine forward

Function	kine_forward(joint_pos)
Description	Calculate the pose value of the specified joint position under the current tool, current installation angle and current user coordinate frame settings
Parameter	joint_pos: Joint space position

Return	success：(0, cartesian_pose) cartesian_pose is like (x,y,z,rx,ry,rz) fail：other

```
'''Kine forward'''
import jkrc
robot = jkrc.RC("192.168.2.64")
 robot.login()
ret = robot.get_joint_position() if ret[0] == 0:
print("the joint position is :",ret[1]) else:
print("some things happend,the errcode is: ",ret[0]) joint_pos = ret[1]
robot.kine_forward(joint_pos) robot.logout()
```

5.16 Robot joint move

Function	joint_move(joint_pos, move_mode ,is_block, speed)
Description	Robot joint move
Parameter	joint_pos: Joint move position move_mode: 0 for absolute move， 1 for incremental move is_block： Set whether the interface is a block interface, TRUE represents a block interface and FALSE represents a non-block interface. speed: Robot joint move speed, unit: rad/s
Return	Success: (0,) fail：other

```
import jkrc
ABS = 0
INCR = 1
robot = jkrc.RC("192.168.2.64")
 robot.login()
robot.joint_move(joint_pos =[0.1,0.1,0.1,0.1,0.1,0.1]                    ,move_mode =
INCR ,is_block = False ,speed = 10 )
ret = robot.get_joint_position() ret_joint_pos = ret[1]
robot.joint_move(joint_pos = ret_joint_pos                    ,move_mode = ABS ,is_block =
False ,speed = 10 )  robot.logout()
```

5.17 Robot end linear move

Function	linear_move(end_pos, move_mode, is_block ,speed)
Description	Robot end linear move
Parameter	end_pos: Robot end move target position move_mode: 0 for absolute move， 1 for incremental move is_block: Set whether the interface is a block interface, TRUE represents a block interface and FALSE represents a non-block interface. speed: Robot linear move speed, unit: mm/s
Return	success：(0,) Fail：other

```
import jkrc
ABS = 0
INCR = 1
robot = jkrc.RC("192.168.2.64")
robot.login()
ret = robot.get_tcp_position() ret_end_pos = ret[1]
robot.linear_move(end_pos = ret_end_pos ,move_mode = ABS,is_block =False,speed
= 50)#
ret = robot.linear_move(end_pos = [453,105,164,1.72,-0.78,-0.07] ,move_mode=
ABS,is_block = False,speed = 50)
print("the return value is :",ret) robot.logout()
```

ps:the example above may not work due to robot types

5.18 Robot servo move enable

Function	servo_move_enable(enable)
Description	Robot servo move enable
Parameter	enable :TRUE means to enter the servo move control mode, FALSE means to quit the mode
Return	success：(0,) fail：other

5.19 Robot joint servo move extension

Function	servo_j(joint_pos, move_mode)

Description	Joint move control mode
Parameter	joint_pos: Joint move position move_mode: Specify move mode:1 for Absolute move and 0 for relative move
Return	Success :(0,) Failed: Other

```
import jkrc
import time
robot = jkrc.RC("192.168.2.64")
ABS = 0
INCR = 1
Enable = True
Disable = False
robot.login()
robot.servo_move_enable(Enable)
for i in range(200):
robot.servo_j(joint_pos =[0.001,0.001,0.001,0.001,0.001,0.001],move_mode=
INCR)#
time.sleep(0.008) for i in range(200):
robot.servo_j(joint_pos=[-0.001,-0.001,-0.001,-0.001,-0.001,-
0.001],move_mode = INCR) time.sleep(0.008)
robot.servo_move_enable(Disable)
robot.logout()
```

5.20 Robot Cartesian servo move

Function	**servo_p(cartesian_pose, move_mode)**
Description	Control mode of robot cartesian space position
Parameter	cartesian_pose: End position of robot cartesian space motion move_mode Specify move mode: 1 stands for absolute move, 0 stands for relative move
Return	success：(0,) fail：other

5.21 Set digital output variables

Function	**set_digital_output(iotype = a_type, index = a_number, value = a_value)**
Description	Set DO Value

Parameter	iotype :DO Type index :DO Index (starting from 0) value :DO Value
Return	success：(0,) fail：other

```
import jkrc
IO_CABINET =0
IO_TOOL = 1
 IO_EXTEND = 2
robot = jkrc.RC("192.168.2.64")# robot.login()
robot.set_digital_output(type = IO_CABINET,index = 2,value = 1)
robot.logout()
```

5.22 Set analog output variables

Function	**set_analog_output(iotype = a_type, index = a_number, value = a_value)**
Description	Set analog output (AO) value
Parameter	iotype :AO Type index :AO Index (starting from 0) value :AO Settings
Return	success：(0,) fail：other

```
 import jkrc
IO_CABINET =0
IO_TOOL = 1
IO_EXTEND = 2
robot = jkrc.RC("192.168.2.64")
robot.login()
robot.set_analog_output(iotype = IO_CABINET,index = 3,value = 1.55) robot.logout()
```

5.23 Get digital input status

Function	**get_digital_input(iotype, index)**
Description	Query DI status
Parameter	iotype :DI Type index :DI Index (starting from 0)
Return	succes：(0,value) failed：other

5.24 Get digital output status

Function	get_digital_output(iotype, index)
Description	Query DO status
Parameter	iotype :DO Type index :DO Index (starting from 0)
Return	success：(0,value) fail：other

5.25 Get analog input variables

Function	get_analog_input(iotype, index)
Description	Get the type of AI value
Parameter	iotype:AI Type index:AI Index (starting from 0)
Return	success：(0,value) failed：other

5.26 Get analog output variables

Function	get_analog_output(type, index)
Description	Get AO value
Parameter	type :AO Type index :AO Index (starting from 0)
Return	success：(0,value) result :Query result of AO status (expressed as a floating point), failed：other

```
import jkrc
IO_CABINET =0
IO_TOOL = 1
IO_EXTEND = 2
robot = jkrc.RC("192.168.2.64")
robot.login()
robot.set_analog_output(iotype = IO_CABINET,index = 3,value = 1.55) ret = robot.get_analog_output(iotype =
IO_CABINET,index = 3) print("AO4 value is:",ret[1])
robot.logout()
```

5.27 Query whether extension IO in running status

Function	is_extio_running()
Description	Query whether the extension IO module is running
Parameter	
Return	success：(0,status) That status is 1 means running fail：other

5.28 Load the specified program

Function	program_load(file_name)
Description	Run the loaded program
Parameter	file_name :program name likes “file_name”
Return	success：(0,) failed：other

5.29 Get the loaded program

Function	get_loaded_program()
Description	Get the name of the loaded operating program
Parameter	
Return	success：(0,file_name) failed：other

```
import jkrc
robot = jkrc.RC("192.168.2.64")
robot.login()
ret = robot.program_load("program_test")
ret = robot.get_loaded_program()
print("the loaded program is:",ret[1]) robot.logout()
```

5.30 Run the loaded program

Function	program_run()
Description	Run the loaded program
Parameter	
Return	success：(0,) failed：other

5.31 Pause the running program

Function	program_pause()
Description	Pause the running program
Parameter	
Return	success：(0,) failed：other

5.32 Resume program

Function	program_resume()
Description	Resume program
Parameter	
Return	success：(0,) failed：other

5.33 Abort program

Function	program_abort()
Description	Abort program
Parameter	
Return	success：(0,) failed：other

5.34 Get program status

Function	get_program_state()
Description	Get the program status
Parameter	
Return	success：(0,state) That state is equal to 0 stands for stoped. That state is equal to 1 stands for running. That state is equal to 2 stands for paused. fail：other

```
import jkrc
import time
PROGRAM_IDLE = 0 PROGRAM_RUNNING =1 PROGRAM_PAUSED = 2
robot = jkrc.RC("192.168.2.64")
 robot.login()
ret = robot.program_load("program_test")
ret = robot.get_loaded_program() print("the loaded program name is:",ret[1]) robot.program_run()
ret = robot.get_program_state()
print("the robot program state is:",ret[1]) time.sleep(10)
robot.program_pause()
time.sleep(10)
robot.program_resume()
time.sleep(10) robot.program_abort() robot.logout()
```

5.35 Set rapid rate

Function	set_rapidrate(rapid_rate)
Description	Set robot rapid rate
Parameter	rapid_rate: The program rapid rate, [0,1]
Return	success：(0,) failed：other

5.36 Get rapid rate

Function	get_rapidrate()
Description	Get robot rapid rate
Parameter	
Return	success：(0, rapidrate) failed：other

5.37 Set tool data

Function	set_tool_data(id,tcp,name)
Description	Set the specified tool
Parameter	id : The range of tool number is [1,10] tcp : [x,y,z,rx,ry,rz] ,Tool coordinate frame is offset relative to flange coordinate frame name: Specify the alias of the tool
Return	success：(0,) failed：other

5.38 Set tool ID

Function	set_tool_id(id)
Description	Set tool ID
Parameter	Id:The value range of tool coordinate frame ID is [0,10], 0 means no tools, flange center
Return	success：(0,) failed：other

5.39 Get the tool ID currently in use

Function	get_tool_id()
Description	Get the tool ID currently in use
Parameter	
Return	success：(0,id) failed：other

5.40 Set user coordinate frame parameter

Function	set_user_frame_data(id, user_frame, name)
Description	Set the parameter of specified user coordinate frame
Parameter	id : The value range of the user coordinate frame number is [1,10] user_frame: Offset value of user coordinate frame[x,y,z,rx,ry,rz] name: Alias of user coordinate frame
Return	success：(0,) failed：other

5.41 Set user coordinate frame ID

Function	set_user_frame_id(id)
Description	Set user coordinate frame ID
Parameter	id: The value range of the user coordinate frame ID is [0,10], where 0 represents the world coordinate frame
Return	success：(0,) failed：other

5.42 Get user coordinate frame ID

Function	get_user_frame_id(id)
Description	Get user coordinate frame ID
Parameter	
Return	success：(0, id) The value range of the id is [0,10], where 0 represents the world coordinate frame failed：other

5.43 Enable drag mode

Function	drag_mode_enable(enable)
Description	Enable drag mode
Parameter	enable :TRUE means to enter the drag mode, FALSE means to quit the drag mode
Return	success：(0,) failed：other

5.44 Query whether in drag mode

Function	is_in_drag_mode()
Description	Query whether in drag mode
Parameter	
Return	Sucess：(0, state) state is equal to 1: the robot is in drag mode. state is equal to 0: the robot is not in drag mode。 Fail: Others

5.45 Get robot status

Function	get_robot_state()
Description	Get robot status
Parameter	
Return	Sucess：(0, state) state is a tuple, whose length is three. The first element of tuple denotes the state of 'emergency stop'. 1: on. 0 :off The second element of tuple denotes the state of 'power supply'. 1: on. 0 :off The third element of tuple denotes the state of 'enable robot'. 1: on. 0 :off Failed: Other

5.46 Query whether on limit

Function	is_on_limit()
Description	Query whether on limit
Parameter	
Return	Sucess： (0, state). state is equal to 1: The robot movement is beyond the limited range state is equal to 0: The robot movement is in the limited range Failed: Other

5.47 Query whether in position

Function	is_in_pos()
Description	Query whether in position
Parameter	

Return	Sucess：(0, state). state is equal to 1: The robot reaches the specified location state is equal to 0: The robot has not reached the specified location. Failed: Other

5.48 Query whether in Collision Protection mode

Function	is_in_collision()
Description	Query whether in Collision Protection mode
Parameter	
Return	Sucess： (0, state). state is equal to 1: the Collision Protection mode is active. state is equal to 0: the Collision Protection mode is not active. Failed: Other

5.49 Clear the robot error code after collision occur

Function	collision_recover()
Discription	Clear the robot error code after collision occur
Parameter	
Return	Success: (0,) Failed: Other

```
import jkrc
robot = jkrc.RC("192.168.2.64")
robot.login()
ret = robot.is_in_collision()# collision_status = ret[1]
if collision_status == 1:
robot.collision_recover()#Clear the robot error after collision happened
else:
print("the robot is not in collision ") robot.logout()
```

5.50 Set collision level

Function	set_collision_level(level)
Description	Set collision level
Parameter	level: the range of collision value is [0,5], 0: close collision, 1: collision threshold 25N, 2: collision threshold 50N, 3: collision threshold 75N, 4: collision threshold 100N, 5: collision threshold 125N

Return	Success: (0,) Failed: Other

5.51 Get collision level

Function	**get_collision_level()**
Description	get collision level
Parameter	
Return	Success: (0,level) level: the collision level. 0: close collision, 1: collision threshold 25N, 2: collision threshold 50N, 3: collision threshold 75N, 4: collision threshold 100N, 5: collision threshold 125N Failed: Other

5.52 Convert Rpy to rot matrix

Function	**rpy_to_rot_matrix(rpy = [rx,ry,rz])**
Description	Convert Rpy to rot matrix
Parameter	rpy: rpy parameters to be converted [rx,ry,rz]
Return	Success: (0, rot_matrix). rot_matrix is a 3X3 rotation matirx Failed: Other

5.53 Convert Rot matrix to rpy

Function	**rot_matrix_to_rpy(rot_matrix)**
Description	Covert Rot matrix to rpy
Parameter	rot_matrix: Rot matrix data to be converted
Return	Success: (0, rpy). rpy is a tuple, whose length is 3. The expression of rpy is (rx, ry, rz). Failed: Other

5.54 Convert Quaternion to rot matrix

Function	quaternion_to_rot_matrix (quaternion = [w,x,y,z])
Description	Convert Quaternion to rot matrix
Parameter	quaternion: Quaternion data to be converted
Return	Success: (0, rot_matrix). rot_matrix is a 3X3 rotation matirx Failed: Other

5.55 Convert Rot matrix to quaternion

Function	rot_matrix_to_quaternion(rot_matrix)
Description	Rot matrix to quaternion
Parameter	rot_matrix：3x3 Rot matrix to be converted
Return	Success: (0, quaternion). quaternion is a tuple, whose length is 4. The expression of quaternion is (w, x, y, z). Failed: Other

5.56 Set callback function in case of a robot error

Function	set_error_handler(func)
Description	Set callback function in case of a robot error
Parameter	func: The function defined by user
Return	Success: (0) Failed: Other

5.57 Torque control enable

Function	torque_control_enable(enable)
Description	Torque control enable
Parameter	enable: TRUE means that the torque control mode is active, FALSE means quit the mode
Return	Success: (0) Failed: Other

5.58 Torque feed forward

Function	torque_feedforward tor_val, grv_flag)
Description	Torque feed forward
Parameter	tor_val : The torque value of each joint for torque feed forward grv_flag: 0 represents the use of the torque feedforward algorithm of the controller, 1 represents the offset value of the joint torque control. 2 means that the joint control torque is completely controlled by the user.
Return	Success: (0) Failed: Other

5.59 Set payload

Function	set_payload(mass = m, centroid = [x,y,z])
Description	Set payload
Parameter	payload Centroid and mass data of payload
Return	Success: (0) Failed: Other

5.60 Get payload data

Function	get_payload()
Description	Get payload data
Parameter	
Return	Success：(0 , payload)　The expression of payload is (mass, (x, y, z)). Failed: Other payload is a tuple,whose length is 2. The first element of tuple is the mass of payload. The second element of tuple is the centroid of payload.

```
import jkrc
robot = jkrc.RC("192.168.2.64")
robot.login()
robot.set_payload(mass= 1,centroid =[0.01,0.02,0.03])# ret = robot.get_payload()
if ret[0] == 0:
print("the payload is :",ret[1]) else:
print("some things happend,the errcode is: ",ret[0]) robot.logout()
```

5.61 Motion abort

Function	motion_abort ()
Description	Terminate the current robotic arm move in any situation
Parameter	
Return	Success：(0 ,) Failed: Other

```
import jkrc
import time
ABS = 0
INCR = 1
robot = jkrc.RC("192.168.2.194")
robot.login()
robot.joint_move(joint_pos=[0.55,0.15,0.15,0.15,0.15,0.15]                    ,move_mode =
INCR ,is_block = False ,speed = 0.05 )
time.sleep(2) robot.motion_abort()
robot.logout()
```

5.62 get robot status

Function	get_robot_status ()
Description	get robot status
Parameter	
Return	Success：(0 , robotstatus) robotstatus have 20 elements as follows： 1. errcode Error code, 0 means normal operation, others represent abnormal operation 2. inpos Whether the robot is in position, 0 means not in position, 1 means in position 3. powered_on Whether the robot is powered on, 0 means not powered on, 1 means powered on 4. enabled Whether the robot is enabled or not, 0 means not enabled, 1 means enabled 5. rapidrate Robot rapid rate 6. protective_stop Whether it has detected a collision, 0 means no collision detected, 1 means collision detected 7. drag_status Whether the robot is in drag status, 0 means not in drag status, 1 means in drag status 8. on_soft_limit Whether the robot is on limit, 0 means limit protection not triggered, 1 means limit protection triggered 9.current_user_id The current user coordinate frame id 10.current_tool_id The current tool id 11. dout Digital output signal of the robot control cabinet, dout[0] is the number of signals 12. din Digital input signal of robot control cabinet, din[0] is the number of signals 13. aout Robot control cabinet analog output signal, aout[0] is the number of signals 14. ain Robot control cabinet analog input signal, ain[0] is the number of signals 15. tio_dout Digital output signal of robot end tool, tio_dout[0] is the number of signals 16. tio_din Digital input signal of robot end tool, tio_din[0] is the number of signals 17. tio_ain Robot control cabinet analog input signal, ain[0] is the number of signals 18. extio The external application digital output signal of the robot, extio[0] is the number of signals 19. cart_position Robot end Cartesian position 20. joint_position Robot joint position

```
import jkrc
import time
robot = jkrc.RC("192.168.2.194")
robot.login()
ret = robot.get_robot_status() print(ret[1])
robot.logout()
```

5.63 Set error code file path

Function	set_errorcode_file_path (errcode_file_path)
Description	Set the error code file path. If you need to use the get_last_error interface, set the error code file path. If no need to use the get_last_error interface, do not set the interface.
Parameter	errcode_file_path：the file path
Return	success：(0 ,) failed：other

5.64 Get the last error code

Function	get_last_error ()
Description	Get the last error code in the robot running process, when clear_error is called, the last error code will be cleared. If you need to use the get_last_error interface, set the error code file path. If no need to use the get_last_error interface, do not set the interface.
Parameter	
Return	success：(0 ,error) failed：other

```
import jkrc
robot = jkrc.RC("192.168.2.194")
robot.login()
robot.program_load("not_exist")#Intentionally loading a non-existent program, causing an error
ret = robot.get_last_error ()#only get error code
print(ret[1])
robot.set_errorcode_file_path("D:\\JAKA_ERROR_CODE.csv")#Paths must be English
ret = robot.get_last_error ()
print(ret[1])
robot.logout()
```

6. Feedback and Error Correct

If there is an inaccurate description or error in the document, the reader is kindly requested to point the finger at criticism. If you find any questions or comments you would like to make during your reading, you can send an email to support@jaka.com and our colleagues will try to respond one by one.

Description of property rights